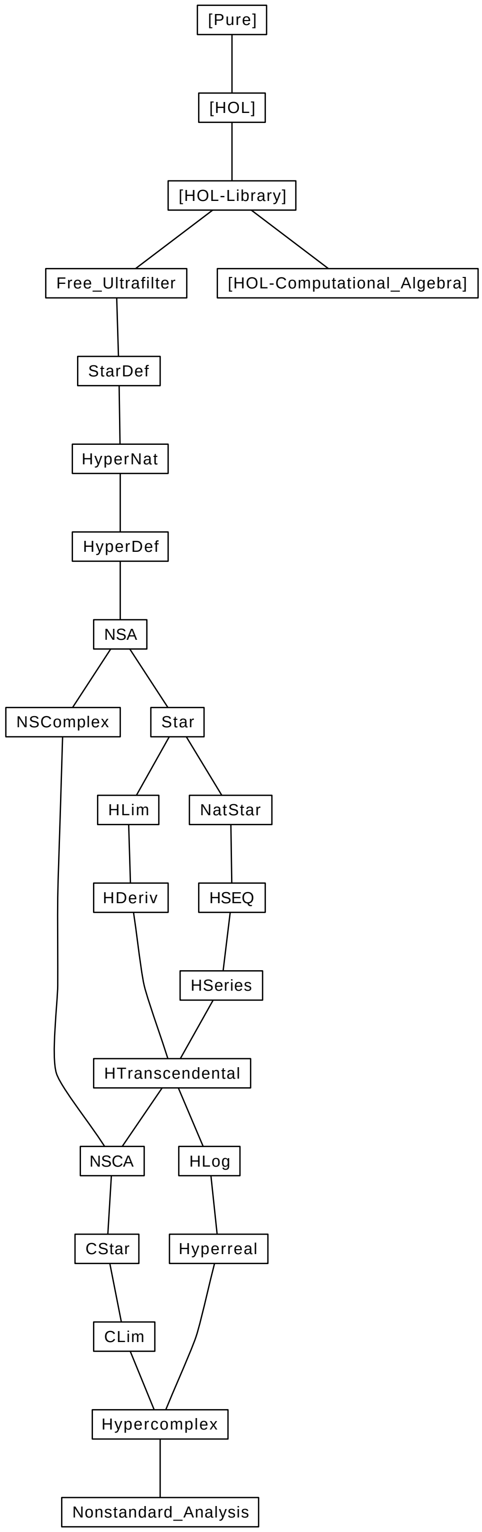
Isabelle/HOL-NSA — Non-Standard Analysis

October 8, 2017

Contents


[Pure]
[HOL]
[HOL-Library]
Free_Ultrafilter
[HOL-Computational_Algebra]
StarDef
HyperNat
HyperDef
NSA
NSComplex
Star
HLim
NatStar
HDeriv
HSEQ
HSeries
HTranscendental
NSCA
HLog
CStar
Hyperreal
CLim
Hypercomplex
Nonstandard_Analysis

1 Filters and Ultrafilters

theory *Free-Ultrafilter*
  imports *HOL−Library.Infinite-Set*
begin

1.1 Definitions and basic properties

1.1.1 Ultrafilters

locale *ultrafilter* =
  fixes F :: *'a filter*
  assumes *proper*: $F \neq bot$
  assumes *ultra*: *eventually* P F ∨ *eventually* $(\lambda x. \neg P\ x)$ F
begin

lemma *eventually-imp-frequently*: *frequently* P F ⟹ *eventually* P F
  ⟨*proof*⟩

lemma *frequently-eq-eventually*: *frequently* P F = *eventually* P F
  ⟨*proof*⟩

lemma *eventually-disj-iff*: *eventually* $(\lambda x. P\ x \vee Q\ x)$ F ⟷ *eventually* P F ∨ *eventually* Q F
  ⟨*proof*⟩

lemma *eventually-all-iff*: *eventually* $(\lambda x. \forall y. P\ x\ y)$ F = $(\forall Y.$ *eventually* $(\lambda x. P\ x\ (Y\ x))$ $F)$
  ⟨*proof*⟩

lemma *eventually-imp-iff*: *eventually* $(\lambda x. P\ x \longrightarrow Q\ x)$ F ⟷ (*eventually* P F ⟶ *eventually* Q F)
  ⟨*proof*⟩

lemma *eventually-iff-iff*: *eventually* $(\lambda x. P\ x \longleftrightarrow Q\ x)$ F ⟷ (*eventually* P F ⟷ *eventually* Q F)
  ⟨*proof*⟩

lemma *eventually-not-iff*: *eventually* $(\lambda x. \neg P\ x)$ F ⟷ ¬ *eventually* P F
  ⟨*proof*⟩

end

1.2 Maximal filter = Ultrafilter

A filter F is an ultrafilter iff it is a maximal filter, i.e. whenever G is a filter and $F \subseteq G$ then $F = G$

Lemma that shows existence of an extension to what was assumed to be a maximal filter. Will be used to derive contradiction in proof of property of ultrafilter.

lemma *extend-filter*: *frequently P F* $\Longrightarrow$ *inf F* (*principal* $\{x.\ P\ x\}$) $\neq$ *bot*
⟨*proof*⟩

lemma *max-filter-ultrafilter*:
assumes $F \neq bot$
assumes *max*: $\bigwedge G.\ G \neq bot \Longrightarrow G \leq F \Longrightarrow F = G$
shows *ultrafilter F*
⟨*proof*⟩

lemma *le-filter-frequently*: $F \leq G \longleftrightarrow (\forall P.$ *frequently P F* $\longrightarrow$ *frequently P G*)
⟨*proof*⟩

lemma (**in** *ultrafilter*) *max-filter*:
assumes *G*: $G \neq bot$
and *sub*: $G \leq F$
shows $F = G$
⟨*proof*⟩

1.3 Ultrafilter Theorem

lemma *ex-max-ultrafilter*:
fixes F :: *'a filter*
assumes *F*: $F \neq bot$
shows $\exists U \leq F.$ *ultrafilter U*
⟨*proof*⟩

1.3.1 Free Ultrafilters

There exists a free ultrafilter on any infinite set.

locale *freeultrafilter* = *ultrafilter* +
assumes *infinite*: *eventually P F* $\Longrightarrow$ *infinite* $\{x.\ P\ x\}$
begin

lemma *finite*: *finite* $\{x.\ P\ x\} \Longrightarrow \neg$ *eventually P F*
⟨*proof*⟩

lemma *finite'*: *finite* $\{x.\ \neg\ P\ x\} \Longrightarrow$ *eventually P F*
⟨*proof*⟩

lemma *le-cofinite*: $F \leq$ *cofinite*
⟨*proof*⟩

lemma *singleton*: $\neg$ *eventually* $(\lambda x.\ x = a)$ *F*
⟨*proof*⟩

lemma *singleton'*: $\neg$ *eventually* $(op = a)$ *F*
⟨*proof*⟩

lemma *ultrafilter*: *ultrafilter F* ⟨*proof*⟩

end

lemma *freeultrafilter-Ex*:
  assumes [*simp*]: *infinite* (*UNIV* :: *'a set*)
  shows ∃ *U*::*'a filter*. *freeultrafilter U*
⟨*proof*⟩

end

2 Construction of Star Types Using Ultrafilters

theory *StarDef*
  imports *Free-Ultrafilter*
begin

2.1 A Free Ultrafilter over the Naturals

definition *FreeUltrafilterNat* :: *nat filter* (𝒰)
  where 𝒰 = (*SOME U*. *freeultrafilter U*)

lemma *freeultrafilter-FreeUltrafilterNat*: *freeultrafilter* 𝒰
  ⟨*proof*⟩

interpretation *FreeUltrafilterNat*: *freeultrafilter* 𝒰
  ⟨*proof*⟩

2.2 Definition of *star* type constructor

definition *starrel* :: ((*nat* ⇒ *'a*) × (*nat* ⇒ *'a*)) *set*
  where *starrel* = {(*X*, *Y*). *eventually* (λ*n*. *X n* = *Y n*) 𝒰}

definition *star* = (*UNIV* :: (*nat* ⇒ *'a*) *set*) // *starrel*

typedef *'a star* = *star* :: (*nat* ⇒ *'a*) *set set*
  ⟨*proof*⟩

definition *star-n* :: (*nat* ⇒ *'a*) ⇒ *'a star*
  where *star-n X* = *Abs-star* (*starrel* `` {*X*})

theorem *star-cases* [*case-names star-n*, *cases type*: *star*]:
  obtains *X* **where** *x* = *star-n X*
  ⟨*proof*⟩

lemma *all-star-eq*: (∀ *x*. *P x*) ⟷ (∀ *X*. *P* (*star-n X*))
  ⟨*proof*⟩

lemma *ex-star-eq*: $(\exists x.\ P\ x) \longleftrightarrow (\exists X.\ P\ (star\text{-}n\ X))$
⟨*proof*⟩

Proving that *starrel* is an equivalence relation.

lemma *starrel-iff* [*iff*]: $(X,\ Y) \in starrel \longleftrightarrow eventually\ (\lambda n.\ X\ n = Y\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *equiv-starrel*: *equiv UNIV starrel*
⟨*proof*⟩

lemmas *equiv-starrel-iff* = *eq-equiv-class-iff* [*OF equiv-starrel UNIV-I UNIV-I*]

lemma *starrel-in-star*: $starrel``\{x\} \in star$
⟨*proof*⟩

lemma *star-n-eq-iff*: $star\text{-}n\ X = star\text{-}n\ Y \longleftrightarrow eventually\ (\lambda n.\ X\ n = Y\ n)\ \mathcal{U}$
⟨*proof*⟩

2.3 Transfer principle

This introduction rule starts each transfer proof.

lemma *transfer-start*: $P \equiv eventually\ (\lambda n.\ Q)\ \mathcal{U} \Longrightarrow Trueprop\ P \equiv Trueprop\ Q$
⟨*proof*⟩

Standard principles that play a central role in the transfer tactic.

definition *Ifun* :: $('a \Rightarrow 'b)\ star \Rightarrow 'a\ star \Rightarrow 'b\ star\ ((\text{-} \star / \text{-})\ [300,\ 301]\ 300)$
where $Ifun\ f \equiv$
$\lambda x.\ Abs\text{-}star\ (\bigcup F{\in}Rep\text{-}star\ f.\ \bigcup X{\in}Rep\text{-}star\ x.\ starrel``\{\lambda n.\ F\ n\ (X\ n)\})$

lemma *Ifun-congruent2*: $congruent2\ starrel\ starrel\ (\lambda F\ X.\ starrel``\{\lambda n.\ F\ n\ (X\ n)\})$
⟨*proof*⟩

lemma *Ifun-star-n*: $star\text{-}n\ F \star star\text{-}n\ X = star\text{-}n\ (\lambda n.\ F\ n\ (X\ n))$
⟨*proof*⟩

lemma *transfer-Ifun*: $f \equiv star\text{-}n\ F \Longrightarrow x \equiv star\text{-}n\ X \Longrightarrow f \star x \equiv star\text{-}n\ (\lambda n.\ F\ n\ (X\ n))$
⟨*proof*⟩

definition *star-of* :: $'a \Rightarrow 'a\ star$
where $star\text{-}of\ x \equiv star\text{-}n\ (\lambda n.\ x)$

Initialize transfer tactic.

⟨*ML*⟩

Transfer introduction rules.

lemma *transfer-ex* [*transfer-intro*]:

$(\bigwedge X.\ p\ (star\text{-}n\ X) \equiv eventually\ (\lambda n.\ P\ n\ (X\ n))\ \mathcal{U}) \Longrightarrow$
$\exists\, x{::}'a\ star.\ p\ x \equiv eventually\ (\lambda n.\ \exists\, x.\ P\ n\ x)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-all* [*transfer-intro*]:
$(\bigwedge X.\ p\ (star\text{-}n\ X) \equiv eventually\ (\lambda n.\ P\ n\ (X\ n))\ \mathcal{U}) \Longrightarrow$
$\forall\, x{::}'a\ star.\ p\ x \equiv eventually\ (\lambda n.\ \forall\, x.\ P\ n\ x)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-not* [*transfer-intro*]: $p \equiv eventually\ P\ \mathcal{U} \Longrightarrow \neg\ p \equiv eventually\ (\lambda n.\ \neg\ P\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-conj* [*transfer-intro*]:
$p \equiv eventually\ P\ \mathcal{U} \Longrightarrow q \equiv eventually\ Q\ \mathcal{U} \Longrightarrow p \wedge q \equiv eventually\ (\lambda n.\ P\ n \wedge Q\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-disj* [*transfer-intro*]:
$p \equiv eventually\ P\ \mathcal{U} \Longrightarrow q \equiv eventually\ Q\ \mathcal{U} \Longrightarrow p \vee q \equiv eventually\ (\lambda n.\ P\ n \vee Q\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-imp* [*transfer-intro*]:
$p \equiv eventually\ P\ \mathcal{U} \Longrightarrow q \equiv eventually\ Q\ \mathcal{U} \Longrightarrow p \longrightarrow q \equiv eventually\ (\lambda n.\ P\ n \longrightarrow Q\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-iff* [*transfer-intro*]:
$p \equiv eventually\ P\ \mathcal{U} \Longrightarrow q \equiv eventually\ Q\ \mathcal{U} \Longrightarrow p = q \equiv eventually\ (\lambda n.\ P\ n = Q\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-if-bool* [*transfer-intro*]:
$p \equiv eventually\ P\ \mathcal{U} \Longrightarrow x \equiv eventually\ X\ \mathcal{U} \Longrightarrow y \equiv eventually\ Y\ \mathcal{U} \Longrightarrow$
$(if\ p\ then\ x\ else\ y) \equiv eventually\ (\lambda n.\ if\ P\ n\ then\ X\ n\ else\ Y\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-eq* [*transfer-intro*]:
$x \equiv star\text{-}n\ X \Longrightarrow y \equiv star\text{-}n\ Y \Longrightarrow x = y \equiv eventually\ (\lambda n.\ X\ n = Y\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-if* [*transfer-intro*]:
$p \equiv eventually\ (\lambda n.\ P\ n)\ \mathcal{U} \Longrightarrow x \equiv star\text{-}n\ X \Longrightarrow y \equiv star\text{-}n\ Y \Longrightarrow$
$(if\ p\ then\ x\ else\ y) \equiv star\text{-}n\ (\lambda n.\ if\ P\ n\ then\ X\ n\ else\ Y\ n)$
⟨*proof*⟩

lemma *transfer-fun-eq* [*transfer-intro*]:
$(\bigwedge X.\ f\ (star\text{-}n\ X) = g\ (star\text{-}n\ X) \equiv eventually\ (\lambda n.\ F\ n\ (X\ n) = G\ n\ (X\ n))$

$\mathcal{U}) \Longrightarrow$
$f = g \equiv$ *eventually* $(\lambda n.\ F\ n = G\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-star-n* [*transfer-intro*]: *star-n* $X \equiv$ *star-n* $(\lambda n.\ X\ n)$
⟨*proof*⟩

lemma *transfer-bool* [*transfer-intro*]: $p \equiv$ *eventually* $(\lambda n.\ p)\ \mathcal{U}$
⟨*proof*⟩

2.4 Standard elements

definition *Standard* :: *'a star set*
where *Standard* = *range star-of*

Transfer tactic should remove occurrences of *star-of*.

⟨*ML*⟩

lemma *star-of-inject*: *star-of* x = *star-of* $y \longleftrightarrow x = y$
⟨*proof*⟩

lemma *Standard-star-of* [*simp*]: *star-of* $x \in$ *Standard*
⟨*proof*⟩

2.5 Internal functions

Transfer tactic should remove occurrences of *Ifun*.

⟨*ML*⟩

lemma *Ifun-star-of* [*simp*]: *star-of* $f \star$ *star-of* x = *star-of* $(f\ x)$
⟨*proof*⟩

lemma *Standard-Ifun* [*simp*]: $f \in$ *Standard* $\Longrightarrow x \in$ *Standard* $\Longrightarrow f \star x \in$ *Standard*
⟨*proof*⟩

Nonstandard extensions of functions.

definition *starfun* :: $('a \Rightarrow 'b) \Rightarrow$ *'a star* $\Rightarrow$ *'b star* $(*f*$ - [*80*] *80*)
where *starfun* $f \equiv \lambda x.$ *star-of* $f \star x$

definition *starfun2* :: $('a \Rightarrow 'b \Rightarrow 'c) \Rightarrow$ *'a star* $\Rightarrow$ *'b star* $\Rightarrow$ *'c star* $(*f2*$ - [*80*] *80*)
where *starfun2* $f \equiv \lambda x\ y.$ *star-of* $f \star x \star y$

declare *starfun-def* [*transfer-unfold*]
declare *starfun2-def* [*transfer-unfold*]

lemma *starfun-star-n*: $(\ *f*\ f)$ (*star-n* X) = *star-n* $(\lambda n.\ f\ (X\ n))$
⟨*proof*⟩

lemma *starfun2-star-n*: ($*f2*$ *f*) (*star-n X*) (*star-n Y*) = *star-n* (λn. *f* (*X n*) (*Y n*))
⟨*proof*⟩

lemma *starfun-star-of* [*simp*]: ($*f*$ *f*) (*star-of x*) = *star-of* (*f x*)
⟨*proof*⟩

lemma *starfun2-star-of* [*simp*]: ($*f2*$ *f*) (*star-of x*) = $*f*$ *f x*
⟨*proof*⟩

lemma *Standard-starfun* [*simp*]: $x \in$ *Standard* $\Longrightarrow$ *starfun f x* $\in$ *Standard*
⟨*proof*⟩

lemma *Standard-starfun2* [*simp*]: $x \in$ *Standard* $\Longrightarrow$ $y \in$ *Standard* $\Longrightarrow$ *starfun2 f x y* $\in$ *Standard*
⟨*proof*⟩

lemma *Standard-starfun-iff*:
assumes *inj*: $\bigwedge x\ y.\ f\ x = f\ y \Longrightarrow x = y$
shows *starfun f x* $\in$ *Standard* $\longleftrightarrow$ $x \in$ *Standard*
⟨*proof*⟩

lemma *Standard-starfun2-iff*:
assumes *inj*: $\bigwedge a\ b\ a'\ b'.\ f\ a\ b = f\ a'\ b' \Longrightarrow a = a' \wedge b = b'$
shows *starfun2 f x y* $\in$ *Standard* $\longleftrightarrow$ $x \in$ *Standard* $\wedge$ $y \in$ *Standard*
⟨*proof*⟩

2.6 Internal predicates

definition *unstar* :: *bool star* $\Rightarrow$ *bool*
where *unstar b* $\longleftrightarrow$ *b* = *star-of True*

lemma *unstar-star-n*: *unstar* (*star-n P*) $\longleftrightarrow$ *eventually P* $\mathcal{U}$
⟨*proof*⟩

lemma *unstar-star-of* [*simp*]: *unstar* (*star-of p*) = *p*
⟨*proof*⟩

Transfer tactic should remove occurrences of *unstar*.

⟨*ML*⟩

lemma *transfer-unstar* [*transfer-intro*]: $p \equiv$ *star-n P* $\Longrightarrow$ *unstar p* $\equiv$ *eventually P* $\mathcal{U}$
⟨*proof*⟩

definition *starP* :: ($'a \Rightarrow$ *bool*) $\Rightarrow$ $'a$ *star* $\Rightarrow$ *bool* ($*p*$ - [*80*] *80*)
where $*p*$ *P* = (λx. *unstar* (*star-of P* $\star$ *x*))

definition *starP2* :: $('a \Rightarrow 'b \Rightarrow bool) \Rightarrow 'a\ star \Rightarrow 'b\ star \Rightarrow bool$ (*∗p2∗* - [*80*] *80*)
  where $*p2*\ P = (\lambda x\ y.\ unstar\ (star\text{-}of\ P \star x \star y))$

declare *starP-def* [*transfer-unfold*]
declare *starP2-def* [*transfer-unfold*]

lemma *starP-star-n*: $(\ *p*\ P)\ (star\text{-}n\ X) = eventually\ (\lambda n.\ P\ (X\ n))\ \mathcal{U}$
  ⟨*proof*⟩

lemma *starP2-star-n*: $(\ *p2*\ P)\ (star\text{-}n\ X)\ (star\text{-}n\ Y) = (eventually\ (\lambda n.\ P\ (X\ n)\ (Y\ n))\ \mathcal{U})$
  ⟨*proof*⟩

lemma *starP-star-of* [*simp*]: $(\ *p*\ P)\ (star\text{-}of\ x) = P\ x$
  ⟨*proof*⟩

lemma *starP2-star-of* [*simp*]: $(\ *p2*\ P)\ (star\text{-}of\ x) = *p*\ P\ x$
  ⟨*proof*⟩

2.7 Internal sets

definition *Iset* :: $'a\ set\ star \Rightarrow 'a\ star\ set$
  where $Iset\ A = \{x.\ (\ *p2*\ op \in)\ x\ A\}$

lemma *Iset-star-n*: $(star\text{-}n\ X \in Iset\ (star\text{-}n\ A)) = (eventually\ (\lambda n.\ X\ n \in A\ n)\ \mathcal{U})$
  ⟨*proof*⟩

Transfer tactic should remove occurrences of *Iset*.

⟨*ML*⟩

lemma *transfer-mem* [*transfer-intro*]:
  $x \equiv star\text{-}n\ X \Longrightarrow a \equiv Iset\ (star\text{-}n\ A) \Longrightarrow x \in a \equiv eventually\ (\lambda n.\ X\ n \in A\ n)\ \mathcal{U}$
  ⟨*proof*⟩

lemma *transfer-Collect* [*transfer-intro*]:
  $(\bigwedge X.\ p\ (star\text{-}n\ X) \equiv eventually\ (\lambda n.\ P\ n\ (X\ n))\ \mathcal{U}) \Longrightarrow$
    $Collect\ p \equiv Iset\ (star\text{-}n\ (\lambda n.\ Collect\ (P\ n)))$
  ⟨*proof*⟩

lemma *transfer-set-eq* [*transfer-intro*]:
  $a \equiv Iset\ (star\text{-}n\ A) \Longrightarrow b \equiv Iset\ (star\text{-}n\ B) \Longrightarrow a = b \equiv eventually\ (\lambda n.\ A\ n = B\ n)\ \mathcal{U}$
  ⟨*proof*⟩

lemma *transfer-ball* [*transfer-intro*]:
  $a \equiv Iset\ (star\text{-}n\ A) \Longrightarrow (\bigwedge X.\ p\ (star\text{-}n\ X) \equiv eventually\ (\lambda n.\ P\ n\ (X\ n))\ \mathcal{U}) \Longrightarrow$

$\forall x{\in}a.\ p\ x \equiv$ *eventually* $(\lambda n.\ \forall x{\in}A\ n.\ P\ n\ x)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-bex* [*transfer-intro*]:
$a \equiv$ *Iset* (*star-n* A) $\Longrightarrow$ ($\bigwedge X.\ p$ (*star-n* X) $\equiv$ *eventually* $(\lambda n.\ P\ n\ (X\ n))\ \mathcal{U}$) $\Longrightarrow$
$\exists x{\in}a.\ p\ x \equiv$ *eventually* $(\lambda n.\ \exists x{\in}A\ n.\ P\ n\ x)\ \mathcal{U}$
⟨*proof*⟩

lemma *transfer-Iset* [*transfer-intro*]: $a \equiv$ *star-n* $A \Longrightarrow$ *Iset* $a \equiv$ *Iset* (*star-n* $(\lambda n.$ $A\ n)$)
⟨*proof*⟩

Nonstandard extensions of sets.

definition *starset* :: *'a set* ⇒ *'a star set* (∗*s*∗ - [*80*] *80*)
where *starset* A = *Iset* (*star-of* A)

declare *starset-def* [*transfer-unfold*]

lemma *starset-mem*: *star-of* $x \in$ ∗*s*∗ $A \longleftrightarrow x \in A$
⟨*proof*⟩

lemma *starset-UNIV*: ∗*s*∗ (*UNIV*::*'a set*) = (*UNIV*::*'a star set*)
⟨*proof*⟩

lemma *starset-empty*: ∗*s*∗ {} = {}
⟨*proof*⟩

lemma *starset-insert*: ∗*s*∗ (*insert* x A) = *insert* (*star-of* x) (∗*s*∗ A)
⟨*proof*⟩

lemma *starset-Un*: ∗*s*∗ $(A \cup B)$ = ∗*s*∗ $A \cup$ ∗*s*∗ B
⟨*proof*⟩

lemma *starset-Int*: ∗*s*∗ $(A \cap B)$ = ∗*s*∗ $A \cap$ ∗*s*∗ B
⟨*proof*⟩

lemma *starset-Compl*: ∗*s*∗ $-A$ = $-$(∗*s*∗ A)
⟨*proof*⟩

lemma *starset-diff*: ∗*s*∗ $(A - B)$ = ∗*s*∗ A − ∗*s*∗ B
⟨*proof*⟩

lemma *starset-image*: ∗*s*∗ (f ‘ A) = (∗*f*∗ f) ‘ (∗*s*∗ A)
⟨*proof*⟩

lemma *starset-vimage*: ∗*s*∗ (f −‘ A) = (∗*f*∗ f) −‘ (∗*s*∗ A)
⟨*proof*⟩

lemma *starset-subset*: (∗*s*∗ $A \subseteq$ ∗*s*∗ B) $\longleftrightarrow A \subseteq B$

⟨*proof*⟩

lemma *starset-eq*: $(*s* A = *s* B) \longleftrightarrow A = B$
⟨*proof*⟩

lemmas *starset-simps* [*simp*] =
starset-mem *starset-UNIV*
starset-empty *starset-insert*
starset-Un *starset-Int*
starset-Compl *starset-diff*
starset-image *starset-vimage*
starset-subset *starset-eq*

2.8 Syntactic classes

instantiation *star* :: (*zero*) *zero*
begin
definition *star-zero-def*: *0* ≡ *star-of 0*
instance ⟨*proof*⟩
end

instantiation *star* :: (*one*) *one*
begin
definition *star-one-def*: *1* ≡ *star-of 1*
instance ⟨*proof*⟩
end

instantiation *star* :: (*plus*) *plus*
begin
definition *star-add-def*: (*op* +) ≡ *f2* (*op* +)
instance ⟨*proof*⟩
end

instantiation *star* :: (*times*) *times*
begin
definition *star-mult-def*: (*op* *) ≡ *f2* (*op* *)
instance ⟨*proof*⟩
end

instantiation *star* :: (*uminus*) *uminus*
begin
definition *star-minus-def*: *uminus* ≡ *f* *uminus*
instance ⟨*proof*⟩
end

instantiation *star* :: (*minus*) *minus*
begin
definition *star-diff-def*: (*op* −) ≡ *f2* (*op* −)
instance ⟨*proof*⟩

end

instantiation *star* :: (*abs*) *abs*
begin
  definition *star-abs-def*: *abs* ≡ ∗*f*∗ *abs*
  instance ⟨*proof*⟩
end

instantiation *star* :: (*sgn*) *sgn*
begin
  definition *star-sgn-def*: *sgn* ≡ ∗*f*∗ *sgn*
  instance ⟨*proof*⟩
end

instantiation *star* :: (*divide*) *divide*
begin
  definition *star-divide-def*: *divide* ≡ ∗*f2*∗ *divide*
  instance ⟨*proof*⟩
end

instantiation *star* :: (*inverse*) *inverse*
begin
  definition *star-inverse-def*: *inverse* ≡ ∗*f*∗ *inverse*
  instance ⟨*proof*⟩
end

instance *star* :: (*Rings.dvd*) *Rings.dvd* ⟨*proof*⟩

instantiation *star* :: (*modulo*) *modulo*
begin
  definition *star-mod-def*: (*op mod*) ≡ ∗*f2*∗ (*op mod*)
  instance ⟨*proof*⟩
end

instantiation *star* :: (*ord*) *ord*
begin
  definition *star-le-def*: (*op* ≤) ≡ ∗*p2*∗ (*op* ≤)
  definition *star-less-def*: (*op* <) ≡ ∗*p2*∗ (*op* <)
  instance ⟨*proof*⟩
end

lemmas *star-class-defs* [*transfer-unfold*] =
  star-zero-def *star-one-def*
  star-add-def *star-diff-def* *star-minus-def*
  star-mult-def *star-divide-def* *star-inverse-def*
  star-le-def *star-less-def* *star-abs-def* *star-sgn-def*
  star-mod-def

Class operations preserve standard elements.

lemma *Standard-zero*: $0 \in Standard$
⟨*proof*⟩

lemma *Standard-one*: $1 \in Standard$
⟨*proof*⟩

lemma *Standard-add*: $x \in Standard \Longrightarrow y \in Standard \Longrightarrow x + y \in Standard$
⟨*proof*⟩

lemma *Standard-diff*: $x \in Standard \Longrightarrow y \in Standard \Longrightarrow x - y \in Standard$
⟨*proof*⟩

lemma *Standard-minus*: $x \in Standard \Longrightarrow - x \in Standard$
⟨*proof*⟩

lemma *Standard-mult*: $x \in Standard \Longrightarrow y \in Standard \Longrightarrow x * y \in Standard$
⟨*proof*⟩

lemma *Standard-divide*: $x \in Standard \Longrightarrow y \in Standard \Longrightarrow x / y \in Standard$
⟨*proof*⟩

lemma *Standard-inverse*: $x \in Standard \Longrightarrow inverse\ x \in Standard$
⟨*proof*⟩

lemma *Standard-abs*: $x \in Standard \Longrightarrow |x| \in Standard$
⟨*proof*⟩

lemma *Standard-mod*: $x \in Standard \Longrightarrow y \in Standard \Longrightarrow x\ mod\ y \in Standard$
⟨*proof*⟩

lemmas *Standard-simps* [*simp*] =
Standard-zero *Standard-one*
Standard-add *Standard-diff* *Standard-minus*
Standard-mult *Standard-divide* *Standard-inverse*
Standard-abs *Standard-mod*

star-of preserves class operations.

lemma *star-of-add*: $star\text{-}of\ (x + y) = star\text{-}of\ x + star\text{-}of\ y$
⟨*proof*⟩

lemma *star-of-diff*: $star\text{-}of\ (x - y) = star\text{-}of\ x - star\text{-}of\ y$
⟨*proof*⟩

lemma *star-of-minus*: $star\text{-}of\ (-x) = - star\text{-}of\ x$
⟨*proof*⟩

lemma *star-of-mult*: $star\text{-}of\ (x * y) = star\text{-}of\ x * star\text{-}of\ y$
⟨*proof*⟩

lemma *star-of-divide*: *star-of* $(x\ /\ y)$ = *star-of* x / *star-of* y
⟨*proof*⟩

lemma *star-of-inverse*: *star-of* (*inverse* x) = *inverse* (*star-of* x)
⟨*proof*⟩

lemma *star-of-mod*: *star-of* (x *mod* y) = *star-of* x *mod* *star-of* y
⟨*proof*⟩

lemma *star-of-abs*: *star-of* $|x|$ = |*star-of* x|
⟨*proof*⟩

star-of preserves numerals.

lemma *star-of-zero*: *star-of* *0* = *0*
⟨*proof*⟩

lemma *star-of-one*: *star-of* *1* = *1*
⟨*proof*⟩

star-of preserves orderings.

lemma *star-of-less*: (*star-of* x < *star-of* y) = $(x < y)$
⟨*proof*⟩

lemma *star-of-le*: (*star-of* x ≤ *star-of* y) = $(x \leq y)$
⟨*proof*⟩

lemma *star-of-eq*: (*star-of* x = *star-of* y) = $(x = y)$
⟨*proof*⟩

As above, for *0*.

lemmas *star-of-0-less* = *star-of-less* [*of 0*, *simplified star-of-zero*]
lemmas *star-of-0-le* = *star-of-le* [*of 0*, *simplified star-of-zero*]
lemmas *star-of-0-eq* = *star-of-eq* [*of 0*, *simplified star-of-zero*]

lemmas *star-of-less-0* = *star-of-less* [*of - 0*, *simplified star-of-zero*]
lemmas *star-of-le-0* = *star-of-le* [*of - 0*, *simplified star-of-zero*]
lemmas *star-of-eq-0* = *star-of-eq* [*of - 0*, *simplified star-of-zero*]

As above, for *1*.

lemmas *star-of-1-less* = *star-of-less* [*of 1*, *simplified star-of-one*]
lemmas *star-of-1-le* = *star-of-le* [*of 1*, *simplified star-of-one*]
lemmas *star-of-1-eq* = *star-of-eq* [*of 1*, *simplified star-of-one*]

lemmas *star-of-less-1* = *star-of-less* [*of - 1*, *simplified star-of-one*]
lemmas *star-of-le-1* = *star-of-le* [*of - 1*, *simplified star-of-one*]
lemmas *star-of-eq-1* = *star-of-eq* [*of - 1*, *simplified star-of-one*]

lemmas *star-of-simps* [*simp*] =

star-of-add *star-of-diff* *star-of-minus*
star-of-mult *star-of-divide* *star-of-inverse*
star-of-mod *star-of-abs*
star-of-zero *star-of-one*
star-of-less *star-of-le* *star-of-eq*
star-of-0-less *star-of-0-le* *star-of-0-eq*
star-of-less-0 *star-of-le-0* *star-of-eq-0*
star-of-1-less *star-of-1-le* *star-of-1-eq*
star-of-less-1 *star-of-le-1* *star-of-eq-1*

2.9 Ordering and lattice classes

instance *star* :: (*order*) *order*
⟨*proof*⟩

instantiation *star* :: (*semilattice-inf*) *semilattice-inf*
begin
definition *star-inf-def* [*transfer-unfold*]: *inf* ≡ ∗*f2*∗ *inf*
instance ⟨*proof*⟩
end

instantiation *star* :: (*semilattice-sup*) *semilattice-sup*
begin
definition *star-sup-def* [*transfer-unfold*]: *sup* ≡ ∗*f2*∗ *sup*
instance ⟨*proof*⟩
end

instance *star* :: (*lattice*) *lattice* ⟨*proof*⟩

instance *star* :: (*distrib-lattice*) *distrib-lattice*
⟨*proof*⟩

lemma *Standard-inf* [*simp*]: *x* ∈ *Standard* ⟹ *y* ∈ *Standard* ⟹ *inf x y* ∈ *Standard*
⟨*proof*⟩

lemma *Standard-sup* [*simp*]: *x* ∈ *Standard* ⟹ *y* ∈ *Standard* ⟹ *sup x y* ∈ *Standard*
⟨*proof*⟩

lemma *star-of-inf* [*simp*]: *star-of* (*inf x y*) = *inf* (*star-of x*) (*star-of y*)
⟨*proof*⟩

lemma *star-of-sup* [*simp*]: *star-of* (*sup x y*) = *sup* (*star-of x*) (*star-of y*)
⟨*proof*⟩

instance *star* :: (*linorder*) *linorder*
⟨*proof*⟩

lemma *star-max-def* [*transfer-unfold*]: *max* = *∗f2∗ max*
⟨*proof*⟩

lemma *star-min-def* [*transfer-unfold*]: *min* = *∗f2∗ min*
⟨*proof*⟩

lemma *Standard-max* [*simp*]: $x \in$ *Standard* $\Longrightarrow$ $y \in$ *Standard* $\Longrightarrow$ *max x y* $\in$ *Standard*
⟨*proof*⟩

lemma *Standard-min* [*simp*]: $x \in$ *Standard* $\Longrightarrow$ $y \in$ *Standard* $\Longrightarrow$ *min x y* $\in$ *Standard*
⟨*proof*⟩

lemma *star-of-max* [*simp*]: *star-of* (*max x y*) = *max* (*star-of x*) (*star-of y*)
⟨*proof*⟩

lemma *star-of-min* [*simp*]: *star-of* (*min x y*) = *min* (*star-of x*) (*star-of y*)
⟨*proof*⟩

2.10 Ordered group classes

instance *star* :: (*semigroup-add*) *semigroup-add*
⟨*proof*⟩

instance *star* :: (*ab-semigroup-add*) *ab-semigroup-add*
⟨*proof*⟩

instance *star* :: (*semigroup-mult*) *semigroup-mult*
⟨*proof*⟩

instance *star* :: (*ab-semigroup-mult*) *ab-semigroup-mult*
⟨*proof*⟩

instance *star* :: (*comm-monoid-add*) *comm-monoid-add*
⟨*proof*⟩

instance *star* :: (*monoid-mult*) *monoid-mult*
⟨*proof*⟩

instance *star* :: (*power*) *power* ⟨*proof*⟩

instance *star* :: (*comm-monoid-mult*) *comm-monoid-mult*
⟨*proof*⟩

instance *star* :: (*cancel-semigroup-add*) *cancel-semigroup-add*
⟨*proof*⟩

instance *star* :: (*cancel-ab-semigroup-add*) *cancel-ab-semigroup-add*

⟨*proof*⟩

instance *star* :: (*cancel-comm-monoid-add*) *cancel-comm-monoid-add* ⟨*proof*⟩

instance *star* :: (*ab-group-add*) *ab-group-add*
⟨*proof*⟩

instance *star* :: (*ordered-ab-semigroup-add*) *ordered-ab-semigroup-add*
⟨*proof*⟩

instance *star* :: (*ordered-cancel-ab-semigroup-add*) *ordered-cancel-ab-semigroup-add*
⟨*proof*⟩

instance *star* :: (*ordered-ab-semigroup-add-imp-le*) *ordered-ab-semigroup-add-imp-le*
⟨*proof*⟩

instance *star* :: (*ordered-comm-monoid-add*) *ordered-comm-monoid-add* ⟨*proof*⟩
instance *star* :: (*ordered-ab-semigroup-monoid-add-imp-le*) *ordered-ab-semigroup-monoid-add-imp-le*
⟨*proof*⟩
instance *star* :: (*ordered-cancel-comm-monoid-add*) *ordered-cancel-comm-monoid-add*
⟨*proof*⟩
instance *star* :: (*ordered-ab-group-add*) *ordered-ab-group-add* ⟨*proof*⟩

instance *star* :: (*ordered-ab-group-add-abs*) *ordered-ab-group-add-abs*
⟨*proof*⟩

instance *star* :: (*linordered-cancel-ab-semigroup-add*) *linordered-cancel-ab-semigroup-add*
⟨*proof*⟩

2.11 Ring and field classes

instance *star* :: (*semiring*) *semiring*
⟨*proof*⟩

instance *star* :: (*semiring-0*) *semiring-0*
⟨*proof*⟩

instance *star* :: (*semiring-0-cancel*) *semiring-0-cancel* ⟨*proof*⟩

instance *star* :: (*comm-semiring*) *comm-semiring*
⟨*proof*⟩

instance *star* :: (*comm-semiring-0*) *comm-semiring-0* ⟨*proof*⟩
instance *star* :: (*comm-semiring-0-cancel*) *comm-semiring-0-cancel* ⟨*proof*⟩

instance *star* :: (*zero-neq-one*) *zero-neq-one*
⟨*proof*⟩

instance *star* :: (*semiring-1*) *semiring-1* ⟨*proof*⟩

instance *star* :: (*comm-semiring-1*) *comm-semiring-1* ⟨*proof*⟩

declare *dvd-def* [*transfer-refold*]

instance *star* :: (*comm-semiring-1-cancel*) *comm-semiring-1-cancel*
⟨*proof*⟩

instance *star* :: (*semiring-no-zero-divisors*) *semiring-no-zero-divisors*
⟨*proof*⟩

instance *star* :: (*semiring-1-no-zero-divisors*) *semiring-1-no-zero-divisors* ⟨*proof*⟩

instance *star* :: (*semiring-no-zero-divisors-cancel*) *semiring-no-zero-divisors-cancel*
⟨*proof*⟩

instance *star* :: (*semiring-1-cancel*) *semiring-1-cancel* ⟨*proof*⟩
instance *star* :: (*ring*) *ring* ⟨*proof*⟩
instance *star* :: (*comm-ring*) *comm-ring* ⟨*proof*⟩
instance *star* :: (*ring-1*) *ring-1* ⟨*proof*⟩
instance *star* :: (*comm-ring-1*) *comm-ring-1* ⟨*proof*⟩
instance *star* :: (*semidom*) *semidom* ⟨*proof*⟩

instance *star* :: (*semidom-divide*) *semidom-divide*
⟨*proof*⟩

instance *star* :: (*semiring-div*) *semiring-div*
⟨*proof*⟩

instance *star* :: (*ring-no-zero-divisors*) *ring-no-zero-divisors* ⟨*proof*⟩
instance *star* :: (*ring-1-no-zero-divisors*) *ring-1-no-zero-divisors* ⟨*proof*⟩
instance *star* :: (*idom*) *idom* ⟨*proof*⟩
instance *star* :: (*idom-divide*) *idom-divide* ⟨*proof*⟩

instance *star* :: (*division-ring*) *division-ring*
⟨*proof*⟩

instance *star* :: (*field*) *field*
⟨*proof*⟩

instance *star* :: (*ordered-semiring*) *ordered-semiring*
⟨*proof*⟩

instance *star* :: (*ordered-cancel-semiring*) *ordered-cancel-semiring* ⟨*proof*⟩

instance *star* :: (*linordered-semiring-strict*) *linordered-semiring-strict*
⟨*proof*⟩

instance *star* :: (*ordered-comm-semiring*) *ordered-comm-semiring*
⟨*proof*⟩

instance *star* :: (*ordered-cancel-comm-semiring*) *ordered-cancel-comm-semiring* ⟨*proof*⟩

instance *star* :: (*linordered-comm-semiring-strict*) *linordered-comm-semiring-strict* ⟨*proof*⟩

instance *star* :: (*ordered-ring*) *ordered-ring* ⟨*proof*⟩

instance *star* :: (*ordered-ring-abs*) *ordered-ring-abs* ⟨*proof*⟩

instance *star* :: (*abs-if*) *abs-if* ⟨*proof*⟩

instance *star* :: (*linordered-ring-strict*) *linordered-ring-strict* ⟨*proof*⟩
instance *star* :: (*ordered-comm-ring*) *ordered-comm-ring* ⟨*proof*⟩

instance *star* :: (*linordered-semidom*) *linordered-semidom* ⟨*proof*⟩

instance *star* :: (*linordered-idom*) *linordered-idom* ⟨*proof*⟩

instance *star* :: (*linordered-field*) *linordered-field* ⟨*proof*⟩

2.12 Power

lemma *star-power-def* [*transfer-unfold*]: (*op* ^) ≡ λ*x n*. (**f** (λ*x*. *x* ^ *n*)) *x* ⟨*proof*⟩

lemma *Standard-power* [*simp*]: *x* ∈ *Standard* ⟹ *x* ^ *n* ∈ *Standard* ⟨*proof*⟩

lemma *star-of-power* [*simp*]: *star-of* (*x* ^ *n*) = *star-of x* ^ *n* ⟨*proof*⟩

2.13 Number classes

instance *star* :: (*numeral*) *numeral* ⟨*proof*⟩

lemma *star-numeral-def* [*transfer-unfold*]: *numeral k* = *star-of* (*numeral k*) ⟨*proof*⟩

lemma *Standard-numeral* [*simp*]: *numeral k* ∈ *Standard* ⟨*proof*⟩

lemma *star-of-numeral* [*simp*]: *star-of* (*numeral k*) = *numeral k* ⟨*proof*⟩

lemma *star-of-nat-def* [*transfer-unfold*]: *of-nat n* = *star-of* (*of-nat n*)

⟨*proof*⟩

lemmas *star-of-compare-numeral* [*simp*] =
star-of-less [*of numeral k*, *simplified star-of-numeral*]
star-of-le [*of numeral k*, *simplified star-of-numeral*]
star-of-eq [*of numeral k*, *simplified star-of-numeral*]
star-of-less [*of - numeral k*, *simplified star-of-numeral*]
star-of-le [*of - numeral k*, *simplified star-of-numeral*]
star-of-eq [*of - numeral k*, *simplified star-of-numeral*]
star-of-less [*of* − *numeral k*, *simplified star-of-numeral*]
star-of-le [*of* − *numeral k*, *simplified star-of-numeral*]
star-of-eq [*of* − *numeral k*, *simplified star-of-numeral*]
star-of-less [*of -* − *numeral k*, *simplified star-of-numeral*]
star-of-le [*of -* − *numeral k*, *simplified star-of-numeral*]
star-of-eq [*of -* − *numeral k*, *simplified star-of-numeral*] **for** *k*

lemma *Standard-of-nat* [*simp*]: *of-nat n* ∈ *Standard*
⟨*proof*⟩

lemma *star-of-of-nat* [*simp*]: *star-of* (*of-nat n*) = *of-nat n*
⟨*proof*⟩

lemma *star-of-int-def* [*transfer-unfold*]: *of-int z* = *star-of* (*of-int z*)
⟨*proof*⟩

lemma *Standard-of-int* [*simp*]: *of-int z* ∈ *Standard*
⟨*proof*⟩

lemma *star-of-of-int* [*simp*]: *star-of* (*of-int z*) = *of-int z*
⟨*proof*⟩

instance *star* :: (*semiring-char-0*) *semiring-char-0*
⟨*proof*⟩

instance *star* :: (*ring-char-0*) *ring-char-0* ⟨*proof*⟩

instance *star* :: (*semiring-parity*) *semiring-parity*
⟨*proof*⟩

instance *star* :: (*semiring-div-parity*) *semiring-div-parity*
⟨*proof*⟩

instantiation *star* :: (*semiring-numeral-div*) *semiring-numeral-div*
begin

definition *divmod-star* :: *num* ⇒ *num* ⇒ *'a star* × *'a star*
where *divmod-star-def*: *divmod-star m n* = (*numeral m div numeral n*, *numeral m mod numeral n*)

definition *divmod-step-star* :: *num* ⇒ *'a star* × *'a star* ⇒ *'a star* × *'a star*
  where *divmod-step-star l qr* =
    (*let* (*q*, *r*) = *qr*
     in if r ≥ *numeral l then* (*2* ∗ *q* + *1*, *r* − *numeral l*) *else* (*2* ∗ *q*, *r*))

instance
⟨*proof*⟩

end

declare *divmod-algorithm-code* [**where** *?'a* = *'a*::*semiring-numeral-div star*, *code*]

2.14 Finite class

lemma *starset-finite*: *finite A* ⟹ ∗*s*∗ *A* = *star-of* ‘ *A*
  ⟨*proof*⟩

instance *star* :: (*finite*) *finite*
  ⟨*proof*⟩

end

3 Hypernatural numbers

theory *HyperNat*
  imports *StarDef*
begin

type-synonym *hypnat* = *nat star*

abbreviation *hypnat-of-nat* :: *nat* ⇒ *nat star*
  where *hypnat-of-nat* ≡ *star-of*

definition *hSuc* :: *hypnat* ⇒ *hypnat*
  where *hSuc-def* [*transfer-unfold*]: *hSuc* = ∗*f*∗ *Suc*

3.1 Properties Transferred from Naturals

lemma *hSuc-not-zero* [*iff*]: ⋀*m*. *hSuc m* ≠ *0*
  ⟨*proof*⟩

lemma *zero-not-hSuc* [*iff*]: ⋀*m*. *0* ≠ *hSuc m*
  ⟨*proof*⟩

lemma *hSuc-hSuc-eq* [*iff*]: ⋀*m n*. *hSuc m* = *hSuc n* ⟷ *m* = *n*
  ⟨*proof*⟩

lemma *zero-less-hSuc* [*iff*]: ⋀*n*. *0* < *hSuc n*
  ⟨*proof*⟩

lemma *hypnat-minus-zero* [*simp*]: $\bigwedge z::hypnat.\ z - z = 0$
⟨*proof*⟩

lemma *hypnat-diff-0-eq-0* [*simp*]: $\bigwedge n::hypnat.\ 0 - n = 0$
⟨*proof*⟩

lemma *hypnat-add-is-0* [*iff*]: $\bigwedge m\ n::hypnat.\ m + n = 0 \longleftrightarrow m = 0 \wedge n = 0$
⟨*proof*⟩

lemma *hypnat-diff-diff-left*: $\bigwedge i\ j\ k::hypnat.\ i - j - k = i - (j + k)$
⟨*proof*⟩

lemma *hypnat-diff-commute*: $\bigwedge i\ j\ k::hypnat.\ i - j - k = i - k - j$
⟨*proof*⟩

lemma *hypnat-diff-add-inverse* [*simp*]: $\bigwedge m\ n::hypnat.\ n + m - n = m$
⟨*proof*⟩

lemma *hypnat-diff-add-inverse2* [*simp*]: $\bigwedge m\ n::hypnat.\ m + n - n = m$
⟨*proof*⟩

lemma *hypnat-diff-cancel* [*simp*]: $\bigwedge k\ m\ n::hypnat.\ (k + m) - (k + n) = m - n$
⟨*proof*⟩

lemma *hypnat-diff-cancel2* [*simp*]: $\bigwedge k\ m\ n::hypnat.\ (m + k) - (n + k) = m - n$
⟨*proof*⟩

lemma *hypnat-diff-add-0* [*simp*]: $\bigwedge m\ n::hypnat.\ n - (n + m) = 0$
⟨*proof*⟩

lemma *hypnat-diff-mult-distrib*: $\bigwedge k\ m\ n::hypnat.\ (m - n) * k = (m * k) - (n * k)$
⟨*proof*⟩

lemma *hypnat-diff-mult-distrib2*: $\bigwedge k\ m\ n::hypnat.\ k * (m - n) = (k * m) - (k * n)$
⟨*proof*⟩

lemma *hypnat-le-zero-cancel* [*iff*]: $\bigwedge n::hypnat.\ n \leq 0 \longleftrightarrow n = 0$
⟨*proof*⟩

lemma *hypnat-mult-is-0* [*simp*]: $\bigwedge m\ n::hypnat.\ m * n = 0 \longleftrightarrow m = 0 \vee n = 0$
⟨*proof*⟩

lemma *hypnat-diff-is-0-eq* [*simp*]: $\bigwedge m\ n::hypnat.\ m - n = 0 \longleftrightarrow m \leq n$
⟨*proof*⟩

lemma *hypnat-not-less0* [*iff*]: $\bigwedge n::hypnat.\ \neg\ n < 0$

⟨*proof*⟩

lemma *hypnat-less-one* [*iff*]: $\bigwedge n::hypnat.\ n < 1 \longleftrightarrow n = 0$
⟨*proof*⟩

lemma *hypnat-add-diff-inverse*: $\bigwedge m\ n::hypnat.\ \neg\ m < n \Longrightarrow n + (m - n) = m$
⟨*proof*⟩

lemma *hypnat-le-add-diff-inverse* [*simp*]: $\bigwedge m\ n::hypnat.\ n \leq m \Longrightarrow n + (m - n) = m$
⟨*proof*⟩

lemma *hypnat-le-add-diff-inverse2* [*simp*]: $\bigwedge m\ n::hypnat.\ n \leq m \Longrightarrow (m - n) + n = m$
⟨*proof*⟩

declare *hypnat-le-add-diff-inverse2* [*OF order-less-imp-le*]

lemma *hypnat-le0* [*iff*]: $\bigwedge n::hypnat.\ 0 \leq n$
⟨*proof*⟩

lemma *hypnat-le-add1* [*simp*]: $\bigwedge x\ n::hypnat.\ x \leq x + n$
⟨*proof*⟩

lemma *hypnat-add-self-le* [*simp*]: $\bigwedge x\ n::hypnat.\ x \leq n + x$
⟨*proof*⟩

lemma *hypnat-add-one-self-less* [*simp*]: $x < x + 1$ **for** $x :: hypnat$
⟨*proof*⟩

lemma *hypnat-neq0-conv* [*iff*]: $\bigwedge n::hypnat.\ n \neq 0 \longleftrightarrow 0 < n$
⟨*proof*⟩

lemma *hypnat-gt-zero-iff*: $0 < n \longleftrightarrow 1 \leq n$ **for** $n :: hypnat$
⟨*proof*⟩

lemma *hypnat-gt-zero-iff2*: $0 < n \longleftrightarrow (\exists\, m.\ n = m + 1)$ **for** $n :: hypnat$
⟨*proof*⟩

lemma *hypnat-add-self-not-less*: $\neg\ x + y < x$ **for** $x\ y :: hypnat$
⟨*proof*⟩

lemma *hypnat-diff-split*: $P\ (a - b) \longleftrightarrow (a < b \longrightarrow P\ 0) \land (\forall\, d.\ a = b + d \longrightarrow P\ d)$
for $a\ b :: hypnat$
— elimination of $-$ on *hypnat*
⟨*proof*⟩

3.2 Properties of the set of embedded natural numbers

lemma *of-nat-eq-star-of* [*simp*]: *of-nat* = *star-of*
⟨*proof*⟩

lemma *Nats-eq-Standard*: (*Nats* :: *nat star set*) = *Standard*
⟨*proof*⟩

lemma *hypnat-of-nat-mem-Nats* [*simp*]: *hypnat-of-nat n* ∈ *Nats*
⟨*proof*⟩

lemma *hypnat-of-nat-one* [*simp*]: *hypnat-of-nat* (*Suc 0*) = *1*
⟨*proof*⟩

lemma *hypnat-of-nat-Suc* [*simp*]: *hypnat-of-nat* (*Suc n*) = *hypnat-of-nat n* + *1*
⟨*proof*⟩

lemma *of-nat-eq-add* [*rule-format*]: ∀ *d*::*hypnat*. *of-nat m* = *of-nat n* + *d* −−> *d* ∈ *range of-nat*
⟨*proof*⟩

lemma *Nats-diff* [*simp*]: *a* ∈ *Nats* ⟹ *b* ∈ *Nats* ⟹ *a* − *b* ∈ *Nats* **for** *a b* :: *hypnat*
⟨*proof*⟩

3.3 Infinite Hypernatural Numbers – *HNatInfinite*

The set of infinite hypernatural numbers.

definition *HNatInfinite* :: *hypnat set*
where *HNatInfinite* = {*n*. *n* ∉ *Nats*}

lemma *Nats-not-HNatInfinite-iff*: *x* ∈ *Nats* ⟷ *x* ∉ *HNatInfinite*
⟨*proof*⟩

lemma *HNatInfinite-not-Nats-iff*: *x* ∈ *HNatInfinite* ⟷ *x* ∉ *Nats*
⟨*proof*⟩

lemma *star-of-neq-HNatInfinite*: *N* ∈ *HNatInfinite* ⟹ *star-of n* ≠ *N*
⟨*proof*⟩

lemma *star-of-Suc-lessI*: ⋀*N*. *star-of n* < *N* ⟹ *star-of* (*Suc n*) ≠ *N* ⟹ *star-of* (*Suc n*) < *N*
⟨*proof*⟩

lemma *star-of-less-HNatInfinite*:
assumes *N*: *N* ∈ *HNatInfinite*
shows *star-of n* < *N*
⟨*proof*⟩

lemma *star-of-le-HNatInfinite*: $N \in HNatInfinite \Longrightarrow star\text{-}of\ n \leq N$
⟨*proof*⟩

3.3.1 Closure Rules

lemma *Nats-less-HNatInfinite*: $x \in Nats \Longrightarrow y \in HNatInfinite \Longrightarrow x < y$
⟨*proof*⟩

lemma *Nats-le-HNatInfinite*: $x \in Nats \Longrightarrow y \in HNatInfinite \Longrightarrow x \leq y$
⟨*proof*⟩

lemma *zero-less-HNatInfinite*: $x \in HNatInfinite \Longrightarrow 0 < x$
⟨*proof*⟩

lemma *one-less-HNatInfinite*: $x \in HNatInfinite \Longrightarrow 1 < x$
⟨*proof*⟩

lemma *one-le-HNatInfinite*: $x \in HNatInfinite \Longrightarrow 1 \leq x$
⟨*proof*⟩

lemma *zero-not-mem-HNatInfinite* [*simp*]: $0 \notin HNatInfinite$
⟨*proof*⟩

lemma *Nats-downward-closed*: $x \in Nats \Longrightarrow y \leq x \Longrightarrow y \in Nats$ **for** $x\ y :: hypnat$
⟨*proof*⟩

lemma *HNatInfinite-upward-closed*: $x \in HNatInfinite \Longrightarrow x \leq y \Longrightarrow y \in HNatInfinite$
⟨*proof*⟩

lemma *HNatInfinite-add*: $x \in HNatInfinite \Longrightarrow x + y \in HNatInfinite$
⟨*proof*⟩

lemma *HNatInfinite-add-one*: $x \in HNatInfinite \Longrightarrow x + 1 \in HNatInfinite$
⟨*proof*⟩

lemma *HNatInfinite-diff*: $x \in HNatInfinite \Longrightarrow y \in Nats \Longrightarrow x - y \in HNatInfinite$
⟨*proof*⟩

lemma *HNatInfinite-is-Suc*: $x \in HNatInfinite \Longrightarrow \exists\, y.\ x = y + 1$ **for** $x :: hypnat$
⟨*proof*⟩

3.4 Existence of an infinite hypernatural number

ω is in fact an infinite hypernatural number $= [<1,\ 2,\ 3,\ \ldots>]$

definition *whn* :: *hypnat*
where *hypnat-omega-def*: $whn = star\text{-}n\ (\lambda n{::}nat.\ n)$

lemma *hypnat-of-nat-neq-whn*: $hypnat\text{-}of\text{-}nat\ n \neq whn$
⟨*proof*⟩

lemma *whn-neq-hypnat-of-nat*: $whn \neq hypnat\text{-}of\text{-}nat\ n$
⟨*proof*⟩

lemma *whn-not-Nats* [*simp*]: $whn \notin Nats$
⟨*proof*⟩

lemma *HNatInfinite-whn* [*simp*]: $whn \in HNatInfinite$
⟨*proof*⟩

lemma *lemma-unbounded-set* [*simp*]: $eventually\ (\lambda n::nat.\ m < n)\ \mathcal{U}$
⟨*proof*⟩

lemma *hypnat-of-nat-eq*: $hypnat\text{-}of\text{-}nat\ m\ = star\text{-}n\ (\lambda n::nat.\ m)$
⟨*proof*⟩

lemma *SHNat-eq*: $Nats = \{n.\ \exists N.\ n = hypnat\text{-}of\text{-}nat\ N\}$
⟨*proof*⟩

lemma *Nats-less-whn*: $n \in Nats \Longrightarrow n < whn$
⟨*proof*⟩

lemma *Nats-le-whn*: $n \in Nats \Longrightarrow n \leq whn$
⟨*proof*⟩

lemma *hypnat-of-nat-less-whn* [*simp*]: $hypnat\text{-}of\text{-}nat\ n < whn$
⟨*proof*⟩

lemma *hypnat-of-nat-le-whn* [*simp*]: $hypnat\text{-}of\text{-}nat\ n \leq whn$
⟨*proof*⟩

lemma *hypnat-zero-less-hypnat-omega* [*simp*]: $0 < whn$
⟨*proof*⟩

lemma *hypnat-one-less-hypnat-omega* [*simp*]: $1 < whn$
⟨*proof*⟩

3.4.1 Alternative characterization of the set of infinite hypernaturals

$HNatInfinite = \{N.\ \forall n{\in}\mathbb{N}.\ n < N\}$

lemma *HNatInfinite-FreeUltrafilterNat-lemma*:
assumes $\forall N::nat.\ eventually\ (\lambda n.\ f\ n \neq N)\ \mathcal{U}$
shows $eventually\ (\lambda n.\ N < f\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *HNatInfinite-iff*: $HNatInfinite = \{N.\ \forall n \in Nats.\ n < N\}$
⟨*proof*⟩

3.4.2 Alternative Characterization of *HNatInfinite* using Free Ultrafilter

lemma *HNatInfinite-FreeUltrafilterNat*:
star-n X ∈ *HNatInfinite* ⟹ ∀ *u*. *eventually* (λ*n*. *u* < *X n*) 𝒰
⟨*proof*⟩

lemma *FreeUltrafilterNat-HNatInfinite*:
∀ *u*. *eventually* (λ*n*. *u* < *X n*) 𝒰 ⟹ *star-n X* ∈ *HNatInfinite*
⟨*proof*⟩

lemma *HNatInfinite-FreeUltrafilterNat-iff*:
(*star-n X* ∈ *HNatInfinite*) = (∀ *u*. *eventually* (λ*n*. *u* < *X n*) 𝒰)
⟨*proof*⟩

3.5 Embedding of the Hypernaturals into other types

definition *of-hypnat* :: *hypnat* ⇒ *'a*::*semiring-1-cancel star*
where *of-hypnat-def* [*transfer-unfold*]: *of-hypnat* = *f* *of-nat*

lemma *of-hypnat-0* [*simp*]: *of-hypnat 0* = *0*
⟨*proof*⟩

lemma *of-hypnat-1* [*simp*]: *of-hypnat 1* = *1*
⟨*proof*⟩

lemma *of-hypnat-hSuc*: ⋀*m*. *of-hypnat* (*hSuc m*) = *1* + *of-hypnat m*
⟨*proof*⟩

lemma *of-hypnat-add* [*simp*]: ⋀*m n*. *of-hypnat* (*m* + *n*) = *of-hypnat m* + *of-hypnat n*
⟨*proof*⟩

lemma *of-hypnat-mult* [*simp*]: ⋀*m n*. *of-hypnat* (*m* * *n*) = *of-hypnat m* * *of-hypnat n*
⟨*proof*⟩

lemma *of-hypnat-less-iff* [*simp*]:
⋀*m n*. *of-hypnat m* < (*of-hypnat n*::*'a*::*linordered-semidom star*) ⟷ *m* < *n*
⟨*proof*⟩

lemma *of-hypnat-0-less-iff* [*simp*]:
⋀*n*. *0* < (*of-hypnat n*::*'a*::*linordered-semidom star*) ⟷ *0* < *n*
⟨*proof*⟩

lemma *of-hypnat-less-0-iff* [*simp*]: ⋀*m*. ¬ (*of-hypnat m*::*'a*::*linordered-semidom star*) < *0*
⟨*proof*⟩

lemma *of-hypnat-le-iff* [*simp*]:

$\bigwedge m\ n.\ \textit{of-hypnat}\ m \le (\textit{of-hypnat}\ n{::}'a{::}\textit{linordered-semidom star}) \longleftrightarrow m \le n$
⟨proof⟩

lemma *of-hypnat-0-le-iff* [*simp*]: $\bigwedge n.\ 0 \le (\textit{of-hypnat}\ n{::}'a{::}\textit{linordered-semidom star})$
⟨proof⟩

lemma *of-hypnat-le-0-iff* [*simp*]: $\bigwedge m.\ (\textit{of-hypnat}\ m{::}'a{::}\textit{linordered-semidom star}) \le 0 \longleftrightarrow m = 0$
⟨proof⟩

lemma *of-hypnat-eq-iff* [*simp*]:
$\bigwedge m\ n.\ \textit{of-hypnat}\ m = (\textit{of-hypnat}\ n{::}'a{::}\textit{linordered-semidom star}) \longleftrightarrow m = n$
⟨proof⟩

lemma *of-hypnat-eq-0-iff* [*simp*]: $\bigwedge m.\ (\textit{of-hypnat}\ m{::}'a{::}\textit{linordered-semidom star}) = 0 \longleftrightarrow m = 0$
⟨proof⟩

lemma *HNatInfinite-of-hypnat-gt-zero*:
$N \in \textit{HNatInfinite} \Longrightarrow (0{::}'a{::}\textit{linordered-semidom star}) < \textit{of-hypnat}\ N$
⟨proof⟩

end

4 Construction of Hyperreals Using Ultrafilters

theory *HyperDef*
imports *Complex-Main HyperNat*
begin

type-synonym *hypreal* = *real star*

abbreviation *hypreal-of-real* :: *real* ⇒ *real star*
where *hypreal-of-real* ≡ *star-of*

abbreviation *hypreal-of-hypnat* :: *hypnat* ⇒ *hypreal*
where *hypreal-of-hypnat* ≡ *of-hypnat*

definition *omega* :: *hypreal* (ω)
where $\omega = \textit{star-n}\ (\lambda n.\ \textit{real}\ (\textit{Suc}\ n))$
— an infinite number = [<*1*, *2*, *3*, ...>]

definition *epsilon* :: *hypreal* (ε)
where $\varepsilon = \textit{star-n}\ (\lambda n.\ \textit{inverse}\ (\textit{real}\ (\textit{Suc}\ n)))$
— an infinitesimal number = [<*1*, *1/2*, *1/3*, ...>]

4.1 Real vector class instances

instantiation *star* :: (*scaleR*) *scaleR*
begin
  definition *star-scaleR-def* [*transfer-unfold*]: *scaleR r* ≡ *∗f∗* (*scaleR r*)
  instance ⟨*proof*⟩
end

lemma *Standard-scaleR* [*simp*]: *x* ∈ *Standard* ⟹ *scaleR r x* ∈ *Standard*
  ⟨*proof*⟩

lemma *star-of-scaleR* [*simp*]: *star-of* (*scaleR r x*) = *scaleR r* (*star-of x*)
  ⟨*proof*⟩

instance *star* :: (*real-vector*) *real-vector*
⟨*proof*⟩

instance *star* :: (*real-algebra*) *real-algebra*
⟨*proof*⟩

instance *star* :: (*real-algebra-1*) *real-algebra-1* ⟨*proof*⟩

instance *star* :: (*real-div-algebra*) *real-div-algebra* ⟨*proof*⟩

instance *star* :: (*field-char-0*) *field-char-0* ⟨*proof*⟩

instance *star* :: (*real-field*) *real-field* ⟨*proof*⟩

lemma *star-of-real-def* [*transfer-unfold*]: *of-real r* = *star-of* (*of-real r*)
  ⟨*proof*⟩

lemma *Standard-of-real* [*simp*]: *of-real r* ∈ *Standard*
  ⟨*proof*⟩

lemma *star-of-of-real* [*simp*]: *star-of* (*of-real r*) = *of-real r*
  ⟨*proof*⟩

lemma *of-real-eq-star-of* [*simp*]: *of-real* = *star-of*
⟨*proof*⟩

lemma *Reals-eq-Standard*: (ℝ :: *hypreal set*) = *Standard*
  ⟨*proof*⟩

4.2 Injection from *hypreal*

definition *of-hypreal* :: *hypreal* ⇒ *'a::real-algebra-1 star*
  where [*transfer-unfold*]: *of-hypreal* = *∗f∗ of-real*

lemma *Standard-of-hypreal* [*simp*]: *r* ∈ *Standard* ⟹ *of-hypreal r* ∈ *Standard*
  ⟨*proof*⟩

lemma *of-hypreal-0* [*simp*]: *of-hypreal 0* = *0*
⟨*proof*⟩

lemma *of-hypreal-1* [*simp*]: *of-hypreal 1* = *1*
⟨*proof*⟩

lemma *of-hypreal-add* [*simp*]: $\bigwedge x\ y.$ *of-hypreal* $(x + y)$ = *of-hypreal* x + *of-hypreal* y
⟨*proof*⟩

lemma *of-hypreal-minus* [*simp*]: $\bigwedge x.$ *of-hypreal* $(- x) = -$ *of-hypreal* x
⟨*proof*⟩

lemma *of-hypreal-diff* [*simp*]: $\bigwedge x\ y.$ *of-hypreal* $(x - y)$ = *of-hypreal* x − *of-hypreal* y
⟨*proof*⟩

lemma *of-hypreal-mult* [*simp*]: $\bigwedge x\ y.$ *of-hypreal* $(x * y)$ = *of-hypreal* x ∗ *of-hypreal* y
⟨*proof*⟩

lemma *of-hypreal-inverse* [*simp*]:
$\bigwedge x.$ *of-hypreal* (*inverse* x) =
inverse (*of-hypreal* x :: *'a*::{*real-div-algebra*, *division-ring*} *star*)
⟨*proof*⟩

lemma *of-hypreal-divide* [*simp*]:
$\bigwedge x\ y.$ *of-hypreal* $(x\ /\ y)$ =
(*of-hypreal* x / *of-hypreal* y :: *'a*::{*real-field*, *field*} *star*)
⟨*proof*⟩

lemma *of-hypreal-eq-iff* [*simp*]: $\bigwedge x\ y.$ (*of-hypreal* x = *of-hypreal* y) = $(x = y)$
⟨*proof*⟩

lemma *of-hypreal-eq-0-iff* [*simp*]: $\bigwedge x.$ (*of-hypreal* x = *0*) = $(x = 0)$
⟨*proof*⟩

4.3 Properties of *starrel*

lemma *lemma-starrel-refl* [*simp*]: $x \in$ *starrel* `` $\{x\}$
⟨*proof*⟩

lemma *starrel-in-hypreal* [*simp*]: *starrel*``$\{x\}$:*star*
⟨*proof*⟩

declare *Abs-star-inject* [*simp*] *Abs-star-inverse* [*simp*]
declare *equiv-starrel* [*THEN eq-equiv-class-iff*, *simp*]

4.4 *hypreal-of-real*: the Injection from *real* to *hypreal*

lemma *inj-star-of*: *inj star-of*
⟨*proof*⟩

lemma *mem-Rep-star-iff*: $X \in \textit{Rep-star}\ x \longleftrightarrow x = \textit{star-n}\ X$
⟨*proof*⟩

lemma *Rep-star-star-n-iff* [*simp*]: $X \in \textit{Rep-star}\ (\textit{star-n}\ Y) \longleftrightarrow \textit{eventually}\ (\lambda n.\ Y\ n = X\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *Rep-star-star-n*: $X \in \textit{Rep-star}\ (\textit{star-n}\ X)$
⟨*proof*⟩

4.5 Properties of *star-n*

lemma *star-n-add*: $\textit{star-n}\ X + \textit{star-n}\ Y = \textit{star-n}\ (\lambda n.\ X\ n + Y\ n)$
⟨*proof*⟩

lemma *star-n-minus*: $- \textit{star-n}\ X = \textit{star-n}\ (\lambda n.\ -(X\ n))$
⟨*proof*⟩

lemma *star-n-diff*: $\textit{star-n}\ X - \textit{star-n}\ Y = \textit{star-n}\ (\lambda n.\ X\ n - Y\ n)$
⟨*proof*⟩

lemma *star-n-mult*: $\textit{star-n}\ X * \textit{star-n}\ Y = \textit{star-n}\ (\lambda n.\ X\ n * Y\ n)$
⟨*proof*⟩

lemma *star-n-inverse*: $\textit{inverse}\ (\textit{star-n}\ X) = \textit{star-n}\ (\lambda n.\ \textit{inverse}\ (X\ n))$
⟨*proof*⟩

lemma *star-n-le*: $\textit{star-n}\ X \leq \textit{star-n}\ Y = \textit{eventually}\ (\lambda n.\ X\ n \leq Y\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *star-n-less*: $\textit{star-n}\ X < \textit{star-n}\ Y = \textit{eventually}\ (\lambda n.\ X\ n < Y\ n)\ \mathcal{U}$
⟨*proof*⟩

lemma *star-n-zero-num*: $0 = \textit{star-n}\ (\lambda n.\ 0)$
⟨*proof*⟩

lemma *star-n-one-num*: $1 = \textit{star-n}\ (\lambda n.\ 1)$
⟨*proof*⟩

lemma *star-n-abs*: $|\textit{star-n}\ X| = \textit{star-n}\ (\lambda n.\ |X\ n|)$
⟨*proof*⟩

lemma *hypreal-omega-gt-zero* [*simp*]: $0 < \omega$
⟨*proof*⟩

4.6 Existence of Infinite Hyperreal Number

Existence of infinite number not corresponding to any real number. Use assumption that member $\mathcal{U}$ is not finite.

A few lemmas first.

lemma *lemma-omega-empty-singleton-disj*:
$\{n{::}nat.\ x = real\ n\} = \{\} \lor (\exists\, y.\ \{n{::}nat.\ x = real\ n\} = \{y\})$
⟨*proof*⟩

lemma *lemma-finite-omega-set*: $finite\ \{n{::}nat.\ x = real\ n\}$
⟨*proof*⟩

lemma *not-ex-hypreal-of-real-eq-omega*: $\nexists\, x.\ hypreal\text{-}of\text{-}real\ x = \omega$
⟨*proof*⟩

lemma *hypreal-of-real-not-eq-omega*: $hypreal\text{-}of\text{-}real\ x \neq \omega$
⟨*proof*⟩

Existence of infinitesimal number also not corresponding to any real number.

lemma *lemma-epsilon-empty-singleton-disj*:
$\{n{::}nat.\ x = inverse(real(Suc\ n))\} = \{\} \lor (\exists\, y.\ \{n{::}nat.\ x = inverse(real(Suc\ n))\} = \{y\})$
⟨*proof*⟩

lemma *lemma-finite-epsilon-set*: $finite\ \{n.\ x = inverse\ (real\ (Suc\ n))\}$
⟨*proof*⟩

lemma *not-ex-hypreal-of-real-eq-epsilon*: $\nexists\, x.\ hypreal\text{-}of\text{-}real\ x = \varepsilon$
⟨*proof*⟩

lemma *hypreal-of-real-not-eq-epsilon*: $hypreal\text{-}of\text{-}real\ x \neq \varepsilon$
⟨*proof*⟩

lemma *hypreal-epsilon-not-zero*: $\varepsilon \neq 0$
⟨*proof*⟩

lemma *hypreal-epsilon-inverse-omega*: $\varepsilon = inverse\ \omega$
⟨*proof*⟩

lemma *hypreal-epsilon-gt-zero*: $0 < \varepsilon$
⟨*proof*⟩

4.7 Absolute Value Function for the Hyperreals

lemma *hrabs-add-less*: $|x| < r \Longrightarrow |y| < s \Longrightarrow |x + y| < r + s$
for $x\ y\ r\ s :: hypreal$
⟨*proof*⟩

lemma *hrabs-less-gt-zero*: $|x| < r \Longrightarrow 0 < r$
  for $x\ r$ :: *hypreal*
  ⟨*proof*⟩

lemma *hrabs-disj*: $|x| = x \lor |x| = -x$
  for x :: *'a::abs-if*
  ⟨*proof*⟩

lemma *hrabs-add-lemma-disj*: $y + - x + (y + - z) = |x + - z| \Longrightarrow y = z \lor x = y$
  for $x\ y\ z$ :: *hypreal*
  ⟨*proof*⟩

4.8 Embedding the Naturals into the Hyperreals

abbreviation *hypreal-of-nat* :: *nat* ⇒ *hypreal*
  where *hypreal-of-nat* ≡ *of-nat*

lemma *SNat-eq*: *Nats* = {*n*. ∃ *N*. *n* = *hypreal-of-nat N*}
  ⟨*proof*⟩

Naturals embedded in hyperreals: is a hyperreal c.f. NS extension.

lemma *hypreal-of-nat*: *hypreal-of-nat m* = *star-n* (λ*n*. *real m*)
  ⟨*proof*⟩

⟨*ML*⟩

4.9 Exponentials on the Hyperreals

lemma *hpowr-0* [*simp*]: r ˆ $0 = (1$::*hypreal*)
  for r :: *hypreal*
  ⟨*proof*⟩

lemma *hpowr-Suc* [*simp*]: r ˆ (*Suc n*) = $r * (r$ ˆ $n)$
  for r :: *hypreal*
  ⟨*proof*⟩

lemma *hrealpow-two*: r ˆ *Suc* (*Suc 0*) = $r * r$
  for r :: *hypreal*
  ⟨*proof*⟩

lemma *hrealpow-two-le* [*simp*]: $0 \le r$ ˆ *Suc* (*Suc 0*)
  for r :: *hypreal*
  ⟨*proof*⟩

lemma *hrealpow-two-le-add-order* [*simp*]: $0 \le u$ ˆ *Suc* (*Suc 0*) + v ˆ *Suc* (*Suc 0*)
  for $u\ v$ :: *hypreal*
  ⟨*proof*⟩

lemma *hrealpow-two-le-add-order2* [*simp*]: *0* ≤ *u* ˆ *Suc* (*Suc 0*) + *v* ˆ *Suc* (*Suc 0*) + *w* ˆ *Suc* (*Suc 0*)
for *u v w* :: *hypreal*
⟨*proof*⟩

lemma *hypreal-add-nonneg-eq-0-iff*: *0* ≤ *x* ⟹ *0* ≤ *y* ⟹ *x* + *y* = *0* ⟷ *x* = *0* ∧ *y* = *0*
for *x y* :: *hypreal*
⟨*proof*⟩

lemma *hypreal-three-squares-add-zero-iff*: *x* ∗ *x* + *y* ∗ *y* + *z* ∗ *z* = *0* ⟷ *x* = *0* ∧ *y* = *0* ∧ *z* = *0*
for *x y z* :: *hypreal*
⟨*proof*⟩

lemma *hrealpow-three-squares-add-zero-iff* [*simp*]:
x ˆ *Suc* (*Suc 0*) + *y* ˆ *Suc* (*Suc 0*) + *z* ˆ *Suc* (*Suc 0*) = *0* ⟷ *x* = *0* ∧ *y* = *0* ∧ *z* = *0*
for *x y z* :: *hypreal*
⟨*proof*⟩

lemma *hrabs-hrealpow-two* [*simp*]: |*x* ˆ *Suc* (*Suc 0*)| = *x* ˆ *Suc* (*Suc 0*)
for *x* :: *hypreal*
⟨*proof*⟩

lemma *two-hrealpow-ge-one* [*simp*]: (*1*::*hypreal*) ≤ *2* ˆ *n*
⟨*proof*⟩

lemma *hrealpow*: *star-n X* ˆ *m* = *star-n* (λ*n*. (*X n*::*real*) ˆ *m*)
⟨*proof*⟩

lemma *hrealpow-sum-square-expand*:
(*x* + *y*) ˆ *Suc* (*Suc 0*) =
x ˆ *Suc* (*Suc 0*) + *y* ˆ *Suc* (*Suc 0*) + (*hypreal-of-nat* (*Suc* (*Suc 0*))) ∗ *x* ∗ *y*
for *x y* :: *hypreal*
⟨*proof*⟩

lemma *power-hypreal-of-real-numeral*:
(*numeral v* :: *hypreal*) ˆ *n* = *hypreal-of-real* ((*numeral v*) ˆ *n*)
⟨*proof*⟩
declare *power-hypreal-of-real-numeral* [*of - numeral w*, *simp*] **for** *w*

lemma *power-hypreal-of-real-neg-numeral*:
(− *numeral v* :: *hypreal*) ˆ *n* = *hypreal-of-real* ((− *numeral v*) ˆ *n*)
⟨*proof*⟩
declare *power-hypreal-of-real-neg-numeral* [*of - numeral w*, *simp*] **for** *w*

4.10 Powers with Hypernatural Exponents

Hypernatural powers of hyperreals.

definition *pow* :: *'a::power star* ⇒ *nat star* ⇒ *'a star* (**infixr** *pow 80*)
where *hyperpow-def* [*transfer-unfold*]: *R pow N* = (**f2* op ^*) *R N*

lemma *Standard-hyperpow* [*simp*]: *r* ∈ *Standard* ⟹ *n* ∈ *Standard* ⟹ *r pow n* ∈ *Standard*
⟨*proof*⟩

lemma *hyperpow*: *star-n X pow star-n Y* = *star-n* (λ*n*. *X n* ^ *Y n*)
⟨*proof*⟩

lemma *hyperpow-zero* [*simp*]: ⋀*n*. (*0*::*'a*::{*power*,*semiring-0*} *star*) *pow* (*n* + (*1*::*hypnat*)) = *0*
⟨*proof*⟩

lemma *hyperpow-not-zero*: ⋀*r n*. *r* ≠ (*0*::*'a*::{*field*} *star*) ⟹ *r pow n* ≠ *0*
⟨*proof*⟩

lemma *hyperpow-inverse*: ⋀*r n*. *r* ≠ (*0*::*'a*::*field star*) ⟹ *inverse* (*r pow n*) = (*inverse r*) *pow n*
⟨*proof*⟩

lemma *hyperpow-hrabs*: ⋀*r n*. |*r*::*'a*::{*linordered-idom*} *star*| *pow n* = |*r pow n*|
⟨*proof*⟩

lemma *hyperpow-add*: ⋀*r n m*. (*r*::*'a*::*monoid-mult star*) *pow* (*n* + *m*) = (*r pow n*) ∗ (*r pow m*)
⟨*proof*⟩

lemma *hyperpow-one* [*simp*]: ⋀*r*. (*r*::*'a*::*monoid-mult star*) *pow* (*1*::*hypnat*) = *r*
⟨*proof*⟩

lemma *hyperpow-two*: ⋀*r*. (*r*::*'a*::*monoid-mult star*) *pow* (*2*::*hypnat*) = *r* ∗ *r*
⟨*proof*⟩

lemma *hyperpow-gt-zero*: ⋀*r n*. (*0*::*'a*::{*linordered-semidom*} *star*) < *r* ⟹ *0* < *r pow n*
⟨*proof*⟩

lemma *hyperpow-ge-zero*: ⋀*r n*. (*0*::*'a*::{*linordered-semidom*} *star*) ≤ *r* ⟹ *0* ≤ *r pow n*
⟨*proof*⟩

lemma *hyperpow-le*: ⋀*x y n*. (*0*::*'a*::{*linordered-semidom*} *star*) < *x* ⟹ *x* ≤ *y* ⟹ *x pow n* ≤ *y pow n*
⟨*proof*⟩

lemma *hyperpow-eq-one* [*simp*]: $\bigwedge n.\ 1\ pow\ n = (1::'a::monoid\text{-}mult\ star)$
⟨*proof*⟩

lemma *hrabs-hyperpow-minus* [*simp*]: $\bigwedge(a::'a::linordered\text{-}idom\ star)\ n.\ |(-a)\ pow\ n| = |a\ pow\ n|$
⟨*proof*⟩

lemma *hyperpow-mult*: $\bigwedge r\ s\ n.\ (r * s::'a::comm\text{-}monoid\text{-}mult\ star)\ pow\ n = (r\ pow\ n) * (s\ pow\ n)$
⟨*proof*⟩

lemma *hyperpow-two-le* [*simp*]: $\bigwedge r.\ (0::'a::\{monoid\text{-}mult,linordered\text{-}ring\text{-}strict\}\ star) \leq r\ pow\ 2$
⟨*proof*⟩

lemma *hrabs-hyperpow-two* [*simp*]:
$|x\ pow\ 2| = (x::'a::\{monoid\text{-}mult,linordered\text{-}ring\text{-}strict\}\ star)\ pow\ 2$
⟨*proof*⟩

lemma *hyperpow-two-hrabs* [*simp*]: $|x::'a::linordered\text{-}idom\ star|\ pow\ 2 = x\ pow\ 2$
⟨*proof*⟩

The precondition could be weakened to $(0::'a) \leq x$.

lemma *hypreal-mult-less-mono*: $u < v \Longrightarrow x < y \Longrightarrow 0 < v \Longrightarrow 0 < x \Longrightarrow u * x < v * y$
for $u\ v\ x\ y :: hypreal$
⟨*proof*⟩

lemma *hyperpow-two-gt-one*: $\bigwedge r::'a::linordered\text{-}semidom\ star.\ 1 < r \Longrightarrow 1 < r\ pow\ 2$
⟨*proof*⟩

lemma *hyperpow-two-ge-one*: $\bigwedge r::'a::linordered\text{-}semidom\ star.\ 1 \leq r \Longrightarrow 1 \leq r\ pow\ 2$
⟨*proof*⟩

lemma *two-hyperpow-ge-one* [*simp*]: $(1::hypreal) \leq 2\ pow\ n$
⟨*proof*⟩

lemma *hyperpow-minus-one2* [*simp*]: $\bigwedge n.\ (-\ 1)\ pow\ (2 * n) = (1::hypreal)$
⟨*proof*⟩

lemma *hyperpow-less-le*: $\bigwedge r\ n\ N.\ (0::hypreal) \leq r \Longrightarrow r \leq 1 \Longrightarrow n < N \Longrightarrow r\ pow\ N \leq r\ pow\ n$
⟨*proof*⟩

lemma *hyperpow-SHNat-le*:
$0 \leq r \Longrightarrow r \leq (1::hypreal) \Longrightarrow N \in HNatInfinite \Longrightarrow \forall n{\in}Nats.\ r\ pow\ N \leq r\ pow\ n$

⟨*proof*⟩

lemma *hyperpow-realpow*: (*hypreal-of-real r*) *pow* (*hypnat-of-nat n*) = *hypreal-of-real* (*r* ˆ *n*)
⟨*proof*⟩

lemma *hyperpow-SReal* [*simp*]: (*hypreal-of-real r*) *pow* (*hypnat-of-nat n*) ∈ ℝ
⟨*proof*⟩

lemma *hyperpow-zero-HNatInfinite* [*simp*]: *N* ∈ *HNatInfinite* ⟹ (*0*::*hypreal*) *pow* *N* = *0*
⟨*proof*⟩

lemma *hyperpow-le-le*: (*0*::*hypreal*) ≤ *r* ⟹ *r* ≤ *1* ⟹ *n* ≤ *N* ⟹ *r pow N* ≤ *r pow n*
⟨*proof*⟩

lemma *hyperpow-Suc-le-self2*: (*0*::*hypreal*) ≤ *r* ⟹ *r* < *1* ⟹ *r pow* (*n* + (*1*::*hypnat*)) ≤ *r*
⟨*proof*⟩

lemma *hyperpow-hypnat-of-nat*: ⋀*x*. *x pow hypnat-of-nat n* = *x* ˆ *n*
⟨*proof*⟩

lemma *of-hypreal-hyperpow*:
⋀*x n*. *of-hypreal* (*x pow n*) = (*of-hypreal x*::′*a*::{*real-algebra-1*} *star*) *pow n*
⟨*proof*⟩

end

5 Infinite Numbers, Infinitesimals, Infinitely Close Relation

theory *NSA*
imports *HyperDef HOL−Library.Lub-Glb*
begin

definition *hnorm* :: ′*a*::*real-normed-vector star* ⇒ *real star*
where [*transfer-unfold*]: *hnorm* = ∗*f*∗ *norm*

definition *Infinitesimal* :: (′*a*::*real-normed-vector*) *star set*
where *Infinitesimal* = {*x*. ∀ *r* ∈ *Reals*. *0* < *r* ⟶ *hnorm x* < *r*}

definition *HFinite* :: (′*a*::*real-normed-vector*) *star set*
where *HFinite* = {*x*. ∃ *r* ∈ *Reals*. *hnorm x* < *r*}

definition *HInfinite* :: (′*a*::*real-normed-vector*) *star set*
where *HInfinite* = {*x*. ∀ *r* ∈ *Reals*. *r* < *hnorm x*}

definition *approx* :: $'a$::*real-normed-vector star* ⇒ $'a$ *star* ⇒ *bool* (**infixl** ≈ *50*)
where $x \approx y \longleftrightarrow x - y \in$ *Infinitesimal*
— the "infinitely close" relation

definition *st* :: *hypreal* ⇒ *hypreal*
where *st* = (λ*x*. *SOME r*. $x \in$ *HFinite* ∧ $r \in \mathbb{R}$ ∧ $r \approx x$)
— the standard part of a hyperreal

definition *monad* :: $'a$::*real-normed-vector star* ⇒ $'a$ *star set*
where *monad* $x = \{y.\ x \approx y\}$

definition *galaxy* :: $'a$::*real-normed-vector star* ⇒ $'a$ *star set*
where *galaxy* $x = \{y.\ (x + -y) \in$ *HFinite*$\}$

lemma *SReal-def*: $\mathbb{R} \equiv \{x.\ \exists\, r.\ x =$ *hypreal-of-real* $r\}$
⟨*proof*⟩

5.1 Nonstandard Extension of the Norm Function

definition *scaleHR* :: *real star* ⇒ $'a$ *star* ⇒ $'a$::*real-normed-vector star*
where [*transfer-unfold*]: *scaleHR* = *starfun2 scaleR*

lemma *Standard-hnorm* [*simp*]: $x \in$ *Standard* ⟹ *hnorm* $x \in$ *Standard*
⟨*proof*⟩

lemma *star-of-norm* [*simp*]: *star-of* (*norm x*) = *hnorm* (*star-of x*)
⟨*proof*⟩

lemma *hnorm-ge-zero* [*simp*]: ⋀*x*::$'a$::*real-normed-vector star*. $0 \le$ *hnorm x*
⟨*proof*⟩

lemma *hnorm-eq-zero* [*simp*]: ⋀*x*::$'a$::*real-normed-vector star*. *hnorm* $x = 0 \longleftrightarrow x = 0$
⟨*proof*⟩

lemma *hnorm-triangle-ineq*: ⋀*x y*::$'a$::*real-normed-vector star*. *hnorm* $(x + y) \le$ *hnorm x* + *hnorm y*
⟨*proof*⟩

lemma *hnorm-triangle-ineq3*: ⋀*x y*::$'a$::*real-normed-vector star*. |*hnorm x* − *hnorm y*| ≤ *hnorm* $(x - y)$
⟨*proof*⟩

lemma *hnorm-scaleR*: ⋀*x*::$'a$::*real-normed-vector star*. *hnorm* $(a *_R x)$ = |*star-of a*| ∗ *hnorm x*
⟨*proof*⟩

lemma *hnorm-scaleHR*: ⋀*a* (*x*::$'a$::*real-normed-vector star*). *hnorm* (*scaleHR a x*)

$= |a| * hnorm\ x$
⟨proof⟩

lemma *hnorm-mult-ineq*: $\bigwedge x\ y::'a::real\text{-}normed\text{-}algebra\ star.\ hnorm\ (x * y) \le hnorm\ x * hnorm\ y$
⟨proof⟩

lemma *hnorm-mult*: $\bigwedge x\ y::'a::real\text{-}normed\text{-}div\text{-}algebra\ star.\ hnorm\ (x * y) = hnorm\ x * hnorm\ y$
⟨proof⟩

lemma *hnorm-hyperpow*: $\bigwedge (x::'a::\{real\text{-}normed\text{-}div\text{-}algebra\}\ star)\ n.\ hnorm\ (x\ pow\ n) = hnorm\ x\ pow\ n$
⟨proof⟩

lemma *hnorm-one* [*simp*]: $hnorm\ (1::'a::real\text{-}normed\text{-}div\text{-}algebra\ star) = 1$
⟨proof⟩

lemma *hnorm-zero* [*simp*]: $hnorm\ (0::'a::real\text{-}normed\text{-}vector\ star) = 0$
⟨proof⟩

lemma *zero-less-hnorm-iff* [*simp*]: $\bigwedge x::'a::real\text{-}normed\text{-}vector\ star.\ 0 < hnorm\ x \longleftrightarrow x \neq 0$
⟨proof⟩

lemma *hnorm-minus-cancel* [*simp*]: $\bigwedge x::'a::real\text{-}normed\text{-}vector\ star.\ hnorm\ (-\ x) = hnorm\ x$
⟨proof⟩

lemma *hnorm-minus-commute*: $\bigwedge a\ b::'a::real\text{-}normed\text{-}vector\ star.\ hnorm\ (a - b) = hnorm\ (b - a)$
⟨proof⟩

lemma *hnorm-triangle-ineq2*: $\bigwedge a\ b::'a::real\text{-}normed\text{-}vector\ star.\ hnorm\ a - hnorm\ b \le hnorm\ (a - b)$
⟨proof⟩

lemma *hnorm-triangle-ineq4*: $\bigwedge a\ b::'a::real\text{-}normed\text{-}vector\ star.\ hnorm\ (a - b) \le hnorm\ a + hnorm\ b$
⟨proof⟩

lemma *abs-hnorm-cancel* [*simp*]: $\bigwedge a::'a::real\text{-}normed\text{-}vector\ star.\ |hnorm\ a| = hnorm\ a$
⟨proof⟩

lemma *hnorm-of-hypreal* [*simp*]: $\bigwedge r.\ hnorm\ (of\text{-}hypreal\ r::'a::real\text{-}normed\text{-}algebra\text{-}1\ star) = |r|$
⟨proof⟩

lemma *nonzero-hnorm-inverse*:
$\bigwedge a::'a::real\text{-}normed\text{-}div\text{-}algebra\ star.\ a \neq 0 \Longrightarrow hnorm\ (inverse\ a) = inverse\ (hnorm\ a)$
⟨*proof*⟩

lemma *hnorm-inverse*:
$\bigwedge a::'a::\{real\text{-}normed\text{-}div\text{-}algebra,\ division\text{-}ring\}\ star.\ hnorm\ (inverse\ a) = inverse\ (hnorm\ a)$
⟨*proof*⟩

lemma *hnorm-divide*: $\bigwedge a\ b::'a::\{real\text{-}normed\text{-}field,\ field\}\ star.\ hnorm\ (a\ /\ b) = hnorm\ a\ /\ hnorm\ b$
⟨*proof*⟩

lemma *hypreal-hnorm-def* [*simp*]: $\bigwedge r::hypreal.\ hnorm\ r = |r|$
⟨*proof*⟩

lemma *hnorm-add-less*:
$\bigwedge(x::'a::real\text{-}normed\text{-}vector\ star)\ y\ r\ s.\ hnorm\ x < r \Longrightarrow hnorm\ y < s \Longrightarrow hnorm\ (x + y) < r + s$
⟨*proof*⟩

lemma *hnorm-mult-less*:
$\bigwedge(x::'a::real\text{-}normed\text{-}algebra\ star)\ y\ r\ s.\ hnorm\ x < r \Longrightarrow hnorm\ y < s \Longrightarrow hnorm\ (x * y) < r * s$
⟨*proof*⟩

lemma *hnorm-scaleHR-less*: $|x| < r \Longrightarrow hnorm\ y < s \Longrightarrow hnorm\ (scaleHR\ x\ y) < r * s$
⟨*proof*⟩

5.2 Closure Laws for the Standard Reals

lemma *Reals-minus-iff* [*simp*]: $-\ x \in \mathbb{R} \longleftrightarrow x \in \mathbb{R}$
⟨*proof*⟩

lemma *Reals-add-cancel*: $x + y \in \mathbb{R} \Longrightarrow y \in \mathbb{R} \Longrightarrow x \in \mathbb{R}$
⟨*proof*⟩

lemma *SReal-hrabs*: $x \in \mathbb{R} \Longrightarrow |x| \in \mathbb{R}$
for x :: *hypreal*
⟨*proof*⟩

lemma *SReal-hypreal-of-real* [*simp*]: $hypreal\text{-}of\text{-}real\ x \in \mathbb{R}$
⟨*proof*⟩

lemma *SReal-divide-numeral*: $r \in \mathbb{R} \Longrightarrow r\ /\ (numeral\ w::hypreal) \in \mathbb{R}$
⟨*proof*⟩

ε is not in Reals because it is an infinitesimal

lemma *SReal-epsilon-not-mem*: $\varepsilon \notin \mathbb{R}$
⟨*proof*⟩

lemma *SReal-omega-not-mem*: $\omega \notin \mathbb{R}$
⟨*proof*⟩

lemma *SReal-UNIV-real*: $\{x.\ hypreal\text{-}of\text{-}real\ x \in \mathbb{R}\} = (UNIV::real\ set)$
⟨*proof*⟩

lemma *SReal-iff*: $x \in \mathbb{R} \longleftrightarrow (\exists\, y.\ x = hypreal\text{-}of\text{-}real\ y)$
⟨*proof*⟩

lemma *hypreal-of-real-image*: $hypreal\text{-}of\text{-}real\ `(UNIV::real\ set) = \mathbb{R}$
⟨*proof*⟩

lemma *inv-hypreal-of-real-image*: $inv\ hypreal\text{-}of\text{-}real\ `\ \mathbb{R} = UNIV$
⟨*proof*⟩

lemma *SReal-hypreal-of-real-image*: $\exists\, x.\ x \in P \Longrightarrow P \subseteq \mathbb{R} \Longrightarrow \exists\, Q.\ P = hypreal\text{-}of\text{-}real$
$`\ Q$
⟨*proof*⟩

lemma *SReal-dense*: $x \in \mathbb{R} \Longrightarrow y \in \mathbb{R} \Longrightarrow x < y \Longrightarrow \exists\, r \in Reals.\ x < r \wedge r <$
y
for $x\ y :: hypreal$
⟨*proof*⟩

Completeness of Reals, but both lemmas are unused.

lemma *SReal-sup-lemma*:
$P \subseteq \mathbb{R} \Longrightarrow (\exists\, x \in P.\ y < x) = (\exists\, X.\ hypreal\text{-}of\text{-}real\ X \in P \wedge y < hypreal\text{-}of\text{-}real$
$X)$
⟨*proof*⟩

lemma *SReal-sup-lemma2*:
$P \subseteq \mathbb{R} \Longrightarrow \exists\, x.\ x \in P \Longrightarrow \exists\, y \in Reals.\ \forall\, x \in P.\ x < y \Longrightarrow$
$(\exists\, X.\ X \in \{w.\ hypreal\text{-}of\text{-}real\ w \in P\}) \wedge$
$(\exists\, Y.\ \forall\, X \in \{w.\ hypreal\text{-}of\text{-}real\ w \in P\}.\ X < Y)$
⟨*proof*⟩

5.3 Set of Finite Elements is a Subring of the Extended Reals

lemma *HFinite-add*: $x \in HFinite \Longrightarrow y \in HFinite \Longrightarrow x + y \in HFinite$
⟨*proof*⟩

lemma *HFinite-mult*: $x \in HFinite \Longrightarrow y \in HFinite \Longrightarrow x * y \in HFinite$
for $x\ y :: 'a::real\text{-}normed\text{-}algebra\ star$
⟨*proof*⟩

lemma *HFinite-scaleHR*: $x \in HFinite \Longrightarrow y \in HFinite \Longrightarrow scaleHR\ x\ y \in HFinite$

⟨*proof*⟩

lemma *HFinite-minus-iff*: $- x \in HFinite \longleftrightarrow x \in HFinite$
⟨*proof*⟩

lemma *HFinite-star-of* [*simp*]: $star\text{-}of\ x \in HFinite$
⟨*proof*⟩

lemma *SReal-subset-HFinite*: $(\mathbb{R}::hypreal\ set) \subseteq HFinite$
⟨*proof*⟩

lemma *HFiniteD*: $x \in HFinite \Longrightarrow \exists\, t \in Reals.\ hnorm\ x < t$
⟨*proof*⟩

lemma *HFinite-hrabs-iff* [*iff*]: $|x| \in HFinite \longleftrightarrow x \in HFinite$
for $x :: hypreal$
⟨*proof*⟩

lemma *HFinite-hnorm-iff* [*iff*]: $hnorm\ x \in HFinite \longleftrightarrow x \in HFinite$
for $x :: hypreal$
⟨*proof*⟩

lemma *HFinite-numeral* [*simp*]: $numeral\ w \in HFinite$
⟨*proof*⟩

As always with numerals, *0* and *1* are special cases.

lemma *HFinite-0* [*simp*]: $0 \in HFinite$
⟨*proof*⟩

lemma *HFinite-1* [*simp*]: $1 \in HFinite$
⟨*proof*⟩

lemma *hrealpow-HFinite*: $x \in HFinite \Longrightarrow x\ \hat{}\ n \in HFinite$
for $x :: 'a::\{real\text{-}normed\text{-}algebra,monoid\text{-}mult\}\ star$
⟨*proof*⟩

lemma *HFinite-bounded*: $x \in HFinite \Longrightarrow y \le x \Longrightarrow 0 \le y \Longrightarrow y \in HFinite$
for $x\ y :: hypreal$
⟨*proof*⟩

5.4 Set of Infinitesimals is a Subring of the Hyperreals

lemma *InfinitesimalI*: $(\bigwedge r.\ r \in \mathbb{R} \Longrightarrow 0 < r \Longrightarrow hnorm\ x < r) \Longrightarrow x \in Infinitesimal$
⟨*proof*⟩

lemma *InfinitesimalD*: $x \in Infinitesimal \Longrightarrow \forall\, r \in Reals.\ 0 < r \longrightarrow hnorm\ x < r$
⟨*proof*⟩

lemma *InfinitesimalI2*: $(\bigwedge r.\ 0 < r \Longrightarrow hnorm\ x < star\text{-}of\ r) \Longrightarrow x \in Infinitesimal$
⟨*proof*⟩

lemma *InfinitesimalD2*: $x \in Infinitesimal \Longrightarrow 0 < r \Longrightarrow hnorm\ x < star\text{-}of\ r$
⟨*proof*⟩

lemma *Infinitesimal-zero* [*iff*]: $0 \in Infinitesimal$
⟨*proof*⟩

lemma *hypreal-sum-of-halves*: $x\ /\ 2 + x\ /\ 2 = x$
for $x :: hypreal$
⟨*proof*⟩

lemma *Infinitesimal-add*: $x \in Infinitesimal \Longrightarrow y \in Infinitesimal \Longrightarrow x + y \in Infinitesimal$
⟨*proof*⟩

lemma *Infinitesimal-minus-iff* [*simp*]: $-\ x \in Infinitesimal \longleftrightarrow x \in Infinitesimal$
⟨*proof*⟩

lemma *Infinitesimal-hnorm-iff*: $hnorm\ x \in Infinitesimal \longleftrightarrow x \in Infinitesimal$
⟨*proof*⟩

lemma *Infinitesimal-hrabs-iff* [*iff*]: $|x| \in Infinitesimal \longleftrightarrow x \in Infinitesimal$
for $x :: hypreal$
⟨*proof*⟩

lemma *Infinitesimal-of-hypreal-iff* [*simp*]:
$(of\text{-}hypreal\ x::'a::real\text{-}normed\text{-}algebra\text{-}1\ star) \in Infinitesimal \longleftrightarrow x \in Infinitesimal$
⟨*proof*⟩

lemma *Infinitesimal-diff*: $x \in Infinitesimal \Longrightarrow y \in Infinitesimal \Longrightarrow x - y \in Infinitesimal$
⟨*proof*⟩

lemma *Infinitesimal-mult*: $x \in Infinitesimal \Longrightarrow y \in Infinitesimal \Longrightarrow x * y \in Infinitesimal$
for $x\ y :: 'a::real\text{-}normed\text{-}algebra\ star$
⟨*proof*⟩

lemma *Infinitesimal-HFinite-mult*: $x \in Infinitesimal \Longrightarrow y \in HFinite \Longrightarrow x * y \in Infinitesimal$
for $x\ y :: 'a::real\text{-}normed\text{-}algebra\ star$
⟨*proof*⟩

lemma *Infinitesimal-HFinite-scaleHR*:
$x \in Infinitesimal \Longrightarrow y \in HFinite \Longrightarrow scaleHR\ x\ y \in Infinitesimal$
⟨*proof*⟩

lemma *Infinitesimal-HFinite-mult2*:
$x \in$ *Infinitesimal* $\Longrightarrow y \in$ *HFinite* $\Longrightarrow y * x \in$ *Infinitesimal*
for $x\ y$ :: *'a::real-normed-algebra star*
⟨*proof*⟩

lemma *Infinitesimal-scaleR2*: $x \in$ *Infinitesimal* $\Longrightarrow a *_R x \in$ *Infinitesimal*
⟨*proof*⟩

lemma *Compl-HFinite*: $-$ *HFinite* $=$ *HInfinite*
⟨*proof*⟩

lemma *HInfinite-inverse-Infinitesimal*: $x \in$ *HInfinite* $\Longrightarrow$ *inverse* $x \in$ *Infinitesimal*
for x :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *HInfiniteI*: $(\bigwedge r.\ r \in \mathbb{R} \Longrightarrow r <$ *hnorm* $x) \Longrightarrow x \in$ *HInfinite*
⟨*proof*⟩

lemma *HInfiniteD*: $x \in$ *HInfinite* $\Longrightarrow r \in \mathbb{R} \Longrightarrow r <$ *hnorm* x
⟨*proof*⟩

lemma *HInfinite-mult*: $x \in$ *HInfinite* $\Longrightarrow y \in$ *HInfinite* $\Longrightarrow x * y \in$ *HInfinite*
for $x\ y$ :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *hypreal-add-zero-less-le-mono*: $r < x \Longrightarrow 0 \le y \Longrightarrow r < x + y$
for $r\ x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *HInfinite-add-ge-zero*: $x \in$ *HInfinite* $\Longrightarrow 0 \le y \Longrightarrow 0 \le x \Longrightarrow x + y \in$ *HInfinite*
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *HInfinite-add-ge-zero2*: $x \in$ *HInfinite* $\Longrightarrow 0 \le y \Longrightarrow 0 \le x \Longrightarrow y + x \in$ *HInfinite*
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *HInfinite-add-gt-zero*: $x \in$ *HInfinite* $\Longrightarrow 0 < y \Longrightarrow 0 < x \Longrightarrow x + y \in$ *HInfinite*
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *HInfinite-minus-iff*: $-\ x \in$ *HInfinite* $\longleftrightarrow x \in$ *HInfinite*
⟨*proof*⟩

lemma *HInfinite-add-le-zero*: $x \in$ *HInfinite* $\Longrightarrow y \le 0 \Longrightarrow x \le 0 \Longrightarrow x + y \in$

HInfinite
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *HInfinite-add-lt-zero*: $x \in HInfinite \Longrightarrow y < 0 \Longrightarrow x < 0 \Longrightarrow x + y \in HInfinite$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *HFinite-sum-squares*:
$a \in HFinite \Longrightarrow b \in HFinite \Longrightarrow c \in HFinite \Longrightarrow a * a + b * b + c * c \in HFinite$
for $a\ b\ c$ :: $'a$::*real-normed-algebra star*
⟨*proof*⟩

lemma *not-Infinitesimal-not-zero*: $x \notin Infinitesimal \Longrightarrow x \neq 0$
⟨*proof*⟩

lemma *not-Infinitesimal-not-zero2*: $x \in HFinite - Infinitesimal \Longrightarrow x \neq 0$
⟨*proof*⟩

lemma *HFinite-diff-Infinitesimal-hrabs*:
$x \in HFinite - Infinitesimal \Longrightarrow |x| \in HFinite - Infinitesimal$
for x :: *hypreal*
⟨*proof*⟩

lemma *hnorm-le-Infinitesimal*: $e \in Infinitesimal \Longrightarrow hnorm\ x \leq e \Longrightarrow x \in Infinitesimal$
⟨*proof*⟩

lemma *hnorm-less-Infinitesimal*: $e \in Infinitesimal \Longrightarrow hnorm\ x < e \Longrightarrow x \in Infinitesimal$
⟨*proof*⟩

lemma *hrabs-le-Infinitesimal*: $e \in Infinitesimal \Longrightarrow |x| \leq e \Longrightarrow x \in Infinitesimal$
for x :: *hypreal*
⟨*proof*⟩

lemma *hrabs-less-Infinitesimal*: $e \in Infinitesimal \Longrightarrow |x| < e \Longrightarrow x \in Infinitesimal$
for x :: *hypreal*
⟨*proof*⟩

lemma *Infinitesimal-interval*:
$e \in Infinitesimal \Longrightarrow e' \in Infinitesimal \Longrightarrow e' < x \Longrightarrow x < e \Longrightarrow x \in Infinitesimal$
for x :: *hypreal*
⟨*proof*⟩

lemma *Infinitesimal-interval2*:

$e \in$ *Infinitesimal* $\Longrightarrow e' \in$ *Infinitesimal* $\Longrightarrow e' \leq x \Longrightarrow x \leq e \Longrightarrow x \in$ *Infinitesimal*
for x :: *hypreal*
⟨*proof*⟩

lemma *lemma-Infinitesimal-hyperpow*: $x \in$ *Infinitesimal* $\Longrightarrow 0 < N \Longrightarrow |x$ *pow* $N| \leq |x|$
for x :: *hypreal*
⟨*proof*⟩

lemma *Infinitesimal-hyperpow*: $x \in$ *Infinitesimal* $\Longrightarrow 0 < N \Longrightarrow x$ *pow* $N \in$ *Infinitesimal*
for x :: *hypreal*
⟨*proof*⟩

lemma *hrealpow-hyperpow-Infinitesimal-iff*:
(x ^ $n \in$ *Infinitesimal*) $\longleftrightarrow x$ *pow* (*hypnat-of-nat* n) $\in$ *Infinitesimal*
⟨*proof*⟩

lemma *Infinitesimal-hrealpow*: $x \in$ *Infinitesimal* $\Longrightarrow 0 < n \Longrightarrow x$ ^ $n \in$ *Infinitesimal*
for x :: *hypreal*
⟨*proof*⟩

lemma *not-Infinitesimal-mult*:
$x \notin$ *Infinitesimal* $\Longrightarrow y \notin$ *Infinitesimal* $\Longrightarrow x * y \notin$ *Infinitesimal*
for x y :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *Infinitesimal-mult-disj*: $x * y \in$ *Infinitesimal* $\Longrightarrow x \in$ *Infinitesimal* $\vee y \in$ *Infinitesimal*
for x y :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *HFinite-Infinitesimal-not-zero*: $x \in$ *HFinite−Infinitesimal* $\Longrightarrow x \neq 0$
⟨*proof*⟩

lemma *HFinite-Infinitesimal-diff-mult*:
$x \in$ *HFinite* − *Infinitesimal* $\Longrightarrow y \in$ *HFinite* − *Infinitesimal* $\Longrightarrow x * y \in$ *HFinite* − *Infinitesimal*
for x y :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *Infinitesimal-subset-HFinite*: *Infinitesimal* $\subseteq$ *HFinite*
⟨*proof*⟩

lemma *Infinitesimal-star-of-mult*: $x \in$ *Infinitesimal* $\Longrightarrow x *$ *star-of* $r \in$ *Infinitesimal*
for x :: *'a::real-normed-algebra star*
⟨*proof*⟩

lemma *Infinitesimal-star-of-mult2*: $x \in \mathit{Infinitesimal} \Longrightarrow \mathit{star\text{-}of}\ r * x \in \mathit{Infinitesimal}$
for $x :: {'a}::\mathit{real\text{-}normed\text{-}algebra\ star}$
⟨*proof*⟩

5.5 The Infinitely Close Relation

lemma *mem-infmal-iff*: $x \in \mathit{Infinitesimal} \longleftrightarrow x \approx 0$
⟨*proof*⟩

lemma *approx-minus-iff*: $x \approx y \longleftrightarrow x - y \approx 0$
⟨*proof*⟩

lemma *approx-minus-iff2*: $x \approx y \longleftrightarrow - y + x \approx 0$
⟨*proof*⟩

lemma *approx-refl* [*iff*]: $x \approx x$
⟨*proof*⟩

lemma *hypreal-minus-distrib1*: $- (y + - x) = x + -y$
for $x\ y :: {'a}::\mathit{ab\text{-}group\text{-}add}$
⟨*proof*⟩

lemma *approx-sym*: $x \approx y \Longrightarrow y \approx x$
⟨*proof*⟩

lemma *approx-trans*: $x \approx y \Longrightarrow y \approx z \Longrightarrow x \approx z$
⟨*proof*⟩

lemma *approx-trans2*: $r \approx x \Longrightarrow s \approx x \Longrightarrow r \approx s$
⟨*proof*⟩

lemma *approx-trans3*: $x \approx r \Longrightarrow x \approx s \Longrightarrow r \approx s$
⟨*proof*⟩

lemma *approx-reorient*: $x \approx y \longleftrightarrow y \approx x$
⟨*proof*⟩

Reorientation simplification procedure: reorients (polymorphic) $0 = x$, $1 = x$, $nnn = x$ provided x isn't 0, 1 or a numeral.

⟨*ML*⟩

lemma *Infinitesimal-approx-minus*: $x - y \in \mathit{Infinitesimal} \longleftrightarrow x \approx y$
⟨*proof*⟩

lemma *approx-monad-iff*: $x \approx y \longleftrightarrow \mathit{monad}\ x = \mathit{monad}\ y$
⟨*proof*⟩

lemma *Infinitesimal-approx*: $x \in \mathit{Infinitesimal} \Longrightarrow y \in \mathit{Infinitesimal} \Longrightarrow x \approx y$

⟨*proof*⟩

lemma *approx-add*: $a \approx b \Longrightarrow c \approx d \Longrightarrow a + c \approx b + d$
⟨*proof*⟩

lemma *approx-minus*: $a \approx b \Longrightarrow - a \approx - b$
⟨*proof*⟩

lemma *approx-minus2*: $- a \approx - b \Longrightarrow a \approx b$
⟨*proof*⟩

lemma *approx-minus-cancel* [*simp*]: $- a \approx - b \longleftrightarrow a \approx b$
⟨*proof*⟩

lemma *approx-add-minus*: $a \approx b \Longrightarrow c \approx d \Longrightarrow a + - c \approx b + - d$
⟨*proof*⟩

lemma *approx-diff*: $a \approx b \Longrightarrow c \approx d \Longrightarrow a - c \approx b - d$
⟨*proof*⟩

lemma *approx-mult1*: $a \approx b \Longrightarrow c \in \mathit{HFinite} \Longrightarrow a * c \approx b * c$
for $a\ b\ c$ :: *'a::real-normed-algebra star*
⟨*proof*⟩

lemma *approx-mult2*: $a \approx b \Longrightarrow c \in \mathit{HFinite} \Longrightarrow c * a \approx c * b$
for $a\ b\ c$ :: *'a::real-normed-algebra star*
⟨*proof*⟩

lemma *approx-mult-subst*: $u \approx v * x \Longrightarrow x \approx y \Longrightarrow v \in \mathit{HFinite} \Longrightarrow u \approx v * y$
for $u\ v\ x\ y$ :: *'a::real-normed-algebra star*
⟨*proof*⟩

lemma *approx-mult-subst2*: $u \approx x * v \Longrightarrow x \approx y \Longrightarrow v \in \mathit{HFinite} \Longrightarrow u \approx y * v$
for $u\ v\ x\ y$ :: *'a::real-normed-algebra star*
⟨*proof*⟩

lemma *approx-mult-subst-star-of*: $u \approx x * \mathit{star\text{-}of}\ v \Longrightarrow x \approx y \Longrightarrow u \approx y * \mathit{star\text{-}of}\ v$
for $u\ x\ y$ :: *'a::real-normed-algebra star*
⟨*proof*⟩

lemma *approx-eq-imp*: $a = b \Longrightarrow a \approx b$
⟨*proof*⟩

lemma *Infinitesimal-minus-approx*: $x \in \mathit{Infinitesimal} \Longrightarrow - x \approx x$
⟨*proof*⟩

lemma *bex-Infinitesimal-iff*: $(\exists\, y \in \mathit{Infinitesimal}.\ x - z = y) \longleftrightarrow x \approx z$
⟨*proof*⟩

lemma *bex-Infinitesimal-iff2*: $(\exists\, y \in \mathit{Infinitesimal}.\ x = z + y) \longleftrightarrow x \approx z$
⟨*proof*⟩

lemma *Infinitesimal-add-approx*: $y \in \mathit{Infinitesimal} \Longrightarrow x + y = z \Longrightarrow x \approx z$
⟨*proof*⟩

lemma *Infinitesimal-add-approx-self*: $y \in \mathit{Infinitesimal} \Longrightarrow x \approx x + y$
⟨*proof*⟩

lemma *Infinitesimal-add-approx-self2*: $y \in \mathit{Infinitesimal} \Longrightarrow x \approx y + x$
⟨*proof*⟩

lemma *Infinitesimal-add-minus-approx-self*: $y \in \mathit{Infinitesimal} \Longrightarrow x \approx x + - y$
⟨*proof*⟩

lemma *Infinitesimal-add-cancel*: $y \in \mathit{Infinitesimal} \Longrightarrow x + y \approx z \Longrightarrow x \approx z$
⟨*proof*⟩

lemma *Infinitesimal-add-right-cancel*: $y \in \mathit{Infinitesimal} \Longrightarrow x \approx z + y \Longrightarrow x \approx z$
⟨*proof*⟩

lemma *approx-add-left-cancel*: $d + b \approx d + c \Longrightarrow b \approx c$
⟨*proof*⟩

lemma *approx-add-right-cancel*: $b + d \approx c + d \Longrightarrow b \approx c$
⟨*proof*⟩

lemma *approx-add-mono1*: $b \approx c \Longrightarrow d + b \approx d + c$
⟨*proof*⟩

lemma *approx-add-mono2*: $b \approx c \Longrightarrow b + a \approx c + a$
⟨*proof*⟩

lemma *approx-add-left-iff* [*simp*]: $a + b \approx a + c \longleftrightarrow b \approx c$
⟨*proof*⟩

lemma *approx-add-right-iff* [*simp*]: $b + a \approx c + a \longleftrightarrow b \approx c$
⟨*proof*⟩

lemma *approx-HFinite*: $x \in \mathit{HFinite} \Longrightarrow x \approx y \Longrightarrow y \in \mathit{HFinite}$
⟨*proof*⟩

lemma *approx-star-of-HFinite*: $x \approx \mathit{star\text{-}of}\ D \Longrightarrow x \in \mathit{HFinite}$
⟨*proof*⟩

lemma *approx-mult-HFinite*: $a \approx b \Longrightarrow c \approx d \Longrightarrow b \in \mathit{HFinite} \Longrightarrow d \in \mathit{HFinite}$
$\Longrightarrow a * c \approx b * d$
for $a\ b\ c\ d$ :: *'a::real-normed-algebra star*

⟨*proof*⟩

lemma *scaleHR-left-diff-distrib*: $\bigwedge a\ b\ x.\ scaleHR\ (a - b)\ x = scaleHR\ a\ x - scaleHR\ b\ x$
⟨*proof*⟩

lemma *approx-scaleR1*: $a \approx star\text{-}of\ b \Longrightarrow c \in HFinite \Longrightarrow scaleHR\ a\ c \approx b *_R c$
⟨*proof*⟩

lemma *approx-scaleR2*: $a \approx b \Longrightarrow c *_R a \approx c *_R b$
⟨*proof*⟩

lemma *approx-scaleR-HFinite*: $a \approx star\text{-}of\ b \Longrightarrow c \approx d \Longrightarrow d \in HFinite \Longrightarrow scaleHR\ a\ c \approx b *_R d$
⟨*proof*⟩

lemma *approx-mult-star-of*: $a \approx star\text{-}of\ b \Longrightarrow c \approx star\text{-}of\ d \Longrightarrow a * c \approx star\text{-}of\ b * star\text{-}of\ d$
for *a c* :: *'a::real-normed-algebra star*
⟨*proof*⟩

lemma *approx-SReal-mult-cancel-zero*: $a \in \mathbb{R} \Longrightarrow a \neq 0 \Longrightarrow a * x \approx 0 \Longrightarrow x \approx 0$
for *a x* :: *hypreal*
⟨*proof*⟩

lemma *approx-mult-SReal1*: $a \in \mathbb{R} \Longrightarrow x \approx 0 \Longrightarrow x * a \approx 0$
for *a x* :: *hypreal*
⟨*proof*⟩

lemma *approx-mult-SReal2*: $a \in \mathbb{R} \Longrightarrow x \approx 0 \Longrightarrow a * x \approx 0$
for *a x* :: *hypreal*
⟨*proof*⟩

lemma *approx-mult-SReal-zero-cancel-iff* [*simp*]: $a \in \mathbb{R} \Longrightarrow a \neq 0 \Longrightarrow a * x \approx 0 \longleftrightarrow x \approx 0$
for *a x* :: *hypreal*
⟨*proof*⟩

lemma *approx-SReal-mult-cancel*: $a \in \mathbb{R} \Longrightarrow a \neq 0 \Longrightarrow a * w \approx a * z \Longrightarrow w \approx z$
for *a w z* :: *hypreal*
⟨*proof*⟩

lemma *approx-SReal-mult-cancel-iff1* [*simp*]: $a \in \mathbb{R} \Longrightarrow a \neq 0 \Longrightarrow a * w \approx a * z \longleftrightarrow w \approx z$
for *a w z* :: *hypreal*
⟨*proof*⟩

lemma *approx-le-bound*: $z \le f \Longrightarrow f \approx g \Longrightarrow g \le z ==> f \approx z$
for z :: *hypreal*
⟨*proof*⟩

lemma *approx-hnorm*: $x \approx y \Longrightarrow$ *hnorm* $x \approx$ *hnorm* y
for $x\ y$ :: *'a::real-normed-vector star*
⟨*proof*⟩

5.6 Zero is the Only Infinitesimal that is also a Real

lemma *Infinitesimal-less-SReal*: $x \in \mathbb{R} \Longrightarrow y \in$ *Infinitesimal* $\Longrightarrow 0 < x \Longrightarrow y < x$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *Infinitesimal-less-SReal2*: $y \in$ *Infinitesimal* $\Longrightarrow \forall r \in$ *Reals*. $0 < r \longrightarrow y < r$
for y :: *hypreal*
⟨*proof*⟩

lemma *SReal-not-Infinitesimal*: $0 < y \Longrightarrow y \in \mathbb{R} ==> y \notin$ *Infinitesimal*
for y :: *hypreal*
⟨*proof*⟩

lemma *SReal-minus-not-Infinitesimal*: $y < 0 \Longrightarrow y \in \mathbb{R} \Longrightarrow y \notin$ *Infinitesimal*
for y :: *hypreal*
⟨*proof*⟩

lemma *SReal-Int-Infinitesimal-zero*: $\mathbb{R}$ *Int Infinitesimal* = {*0::hypreal*}
⟨*proof*⟩

lemma *SReal-Infinitesimal-zero*: $x \in \mathbb{R} \Longrightarrow x \in$ *Infinitesimal* $\Longrightarrow x = 0$
for x :: *hypreal*
⟨*proof*⟩

lemma *SReal-HFinite-diff-Infinitesimal*: $x \in \mathbb{R} \Longrightarrow x \neq 0 \Longrightarrow x \in$ *HFinite* $-$ *Infinitesimal*
for x :: *hypreal*
⟨*proof*⟩

lemma *hypreal-of-real-HFinite-diff-Infinitesimal*:
hypreal-of-real $x \neq 0 \Longrightarrow$ *hypreal-of-real* $x \in$ *HFinite* $-$ *Infinitesimal*
⟨*proof*⟩

lemma *star-of-Infinitesimal-iff-0* [*iff*]: *star-of* $x \in$ *Infinitesimal* $\longleftrightarrow x = 0$
⟨*proof*⟩

lemma *star-of-HFinite-diff-Infinitesimal*: $x \neq 0 \Longrightarrow$ *star-of* $x \in$ *HFinite* $-$ *Infinitesimal*
⟨*proof*⟩

lemma *numeral-not-Infinitesimal* [*simp*]:
 $numeral\ w \neq (0::hypreal) \Longrightarrow (numeral\ w :: hypreal) \notin Infinitesimal$
 ⟨*proof*⟩

Again: *1* is a special case, but not *0* this time.

lemma *one-not-Infinitesimal* [*simp*]:
 $(1::'a::\{real\text{-}normed\text{-}vector,zero\text{-}neq\text{-}one\}\ star) \notin Infinitesimal$
 ⟨*proof*⟩

lemma *approx-SReal-not-zero*: $y \in \mathbb{R} \Longrightarrow x \approx y \Longrightarrow y \neq 0 \Longrightarrow x \neq 0$
 for *x y* :: *hypreal*
 ⟨*proof*⟩

lemma *HFinite-diff-Infinitesimal-approx*:
 $x \approx y \Longrightarrow y \in HFinite - Infinitesimal \Longrightarrow x \in HFinite - Infinitesimal$
 ⟨*proof*⟩

The premise $y \neq 0$ is essential; otherwise $x\ /\ y = 0$ and we lose the *HFinite* premise.

lemma *Infinitesimal-ratio*:
 $y \neq 0 \Longrightarrow y \in Infinitesimal \Longrightarrow x\ /\ y \in HFinite \Longrightarrow x \in Infinitesimal$
 for *x y* :: *'a::{real-normed-div-algebra,field} star*
 ⟨*proof*⟩

lemma *Infinitesimal-SReal-divide*: $x \in Infinitesimal \Longrightarrow y \in \mathbb{R} \Longrightarrow x\ /\ y \in Infinitesimal$
 for *x y* :: *hypreal*
 ⟨*proof*⟩

6 Standard Part Theorem

Every finite $x \in R*$ is infinitely close to a unique real number (i.e. a member of *Reals*).

6.1 Uniqueness: Two Infinitely Close Reals are Equal

lemma *star-of-approx-iff* [*simp*]: $star\text{-}of\ x \approx star\text{-}of\ y \longleftrightarrow x = y$
 ⟨*proof*⟩

lemma *SReal-approx-iff*: $x \in \mathbb{R} \Longrightarrow y \in \mathbb{R} \Longrightarrow x \approx y \longleftrightarrow x = y$
 for *x y* :: *hypreal*
 ⟨*proof*⟩

lemma *numeral-approx-iff* [*simp*]:
 $(numeral\ v \approx (numeral\ w :: 'a::\{numeral,real\text{-}normed\text{-}vector\}\ star)) =$
 $(numeral\ v = (numeral\ w :: 'a))$

⟨*proof*⟩

And also for $0 \approx \#nn$ and $1 \approx \#nn$, $\#nn \approx 0$ and $\#nn \approx 1$.

lemma [*simp*]:
$(numeral\ w \approx (0::'a::\{numeral,real\text{-}normed\text{-}vector\}\ star)) = (numeral\ w = (0::'a))$
$((0::'a::\{numeral,real\text{-}normed\text{-}vector\}\ star) \approx numeral\ w) = (numeral\ w = (0::'a))$
$(numeral\ w \approx (1::'b::\{numeral,one,real\text{-}normed\text{-}vector\}\ star)) = (numeral\ w = (1::'b))$
$((1::'b::\{numeral,one,real\text{-}normed\text{-}vector\}\ star) \approx numeral\ w) = (numeral\ w = (1::'b))$
$\neg\ (0 \approx (1::'c::\{zero\text{-}neq\text{-}one,real\text{-}normed\text{-}vector\}\ star))$
$\neg\ (1 \approx (0::'c::\{zero\text{-}neq\text{-}one,real\text{-}normed\text{-}vector\}\ star))$
⟨*proof*⟩

lemma *star-of-approx-numeral-iff* [*simp*]: $star\text{-}of\ k \approx numeral\ w \longleftrightarrow k = numeral\ w$
⟨*proof*⟩

lemma *star-of-approx-zero-iff* [*simp*]: $star\text{-}of\ k \approx 0 \longleftrightarrow k = 0$
⟨*proof*⟩

lemma *star-of-approx-one-iff* [*simp*]: $star\text{-}of\ k \approx 1 \longleftrightarrow k = 1$
⟨*proof*⟩

lemma *approx-unique-real*: $r \in \mathbb{R} \Longrightarrow s \in \mathbb{R} \Longrightarrow r \approx x \Longrightarrow s \approx x \Longrightarrow r = s$
for $r\ s$ :: *hypreal*
⟨*proof*⟩

6.2 Existence of Unique Real Infinitely Close

6.2.1 Lifting of the Ub and Lub Properties

lemma *hypreal-of-real-isUb-iff*: $isUb\ \mathbb{R}\ (hypreal\text{-}of\text{-}real\ `\ Q)\ (hypreal\text{-}of\text{-}real\ Y) = isUb\ UNIV\ Q\ Y$
for Q :: *real set* **and** Y :: *real*
⟨*proof*⟩

lemma *hypreal-of-real-isLub1*: $isLub\ \mathbb{R}\ (hypreal\text{-}of\text{-}real\ `\ Q)\ (hypreal\text{-}of\text{-}real\ Y) \Longrightarrow isLub\ UNIV\ Q\ Y$
for Q :: *real set* **and** Y :: *real*
⟨*proof*⟩

lemma *hypreal-of-real-isLub2*: $isLub\ UNIV\ Q\ Y \Longrightarrow isLub\ \mathbb{R}\ (hypreal\text{-}of\text{-}real\ `\ Q)\ (hypreal\text{-}of\text{-}real\ Y)$
for Q :: *real set* **and** Y :: *real*
⟨*proof*⟩

lemma *hypreal-of-real-isLub-iff*:
$isLub\ \mathbb{R}\ (hypreal\text{-}of\text{-}real\ `\ Q)\ (hypreal\text{-}of\text{-}real\ Y) = isLub\ (UNIV :: real\ set)\ Q\ Y$

for Q :: *real set* **and** Y :: *real*
⟨*proof*⟩

lemma *lemma-isUb-hypreal-of-real*: *isUb* $\mathbb{R}$ P Y $\Longrightarrow \exists$ *Yo*. *isUb* $\mathbb{R}$ P (*hypreal-of-real* *Yo*)
⟨*proof*⟩

lemma *lemma-isLub-hypreal-of-real*: *isLub* $\mathbb{R}$ P Y $\Longrightarrow \exists$ *Yo*. *isLub* $\mathbb{R}$ P (*hypreal-of-real* *Yo*)
⟨*proof*⟩

lemma *lemma-isLub-hypreal-of-real2*: $\exists$ *Yo*. *isLub* $\mathbb{R}$ P (*hypreal-of-real* *Yo*) $\Longrightarrow$ $\exists$ Y. *isLub* $\mathbb{R}$ P Y
⟨*proof*⟩

lemma *SReal-complete*: $P \subseteq \mathbb{R} \Longrightarrow \exists x.\ x \in P \Longrightarrow \exists Y.$ *isUb* $\mathbb{R}$ P Y $\Longrightarrow$ $\exists$ t::*hypreal*. *isLub* $\mathbb{R}$ P t
⟨*proof*⟩

Lemmas about lubs.

lemma *lemma-st-part-ub*: $x \in$ *HFinite* $\Longrightarrow \exists u.$ *isUb* $\mathbb{R}$ $\{s.\ s \in \mathbb{R} \wedge s < x\}$ u
for x :: *hypreal*
⟨*proof*⟩

lemma *lemma-st-part-nonempty*: $x \in$ *HFinite* $\Longrightarrow \exists y.\ y \in \{s.\ s \in \mathbb{R} \wedge s < x\}$
for x :: *hypreal*
⟨*proof*⟩

lemma *lemma-st-part-lub*: $x \in$ *HFinite* $\Longrightarrow \exists t.$ *isLub* $\mathbb{R}$ $\{s.\ s \in \mathbb{R} \wedge s < x\}$ t
for x :: *hypreal*
⟨*proof*⟩

lemma *lemma-st-part-le1*:
$x \in$ *HFinite* $\Longrightarrow$ *isLub* $\mathbb{R}$ $\{s.\ s \in \mathbb{R} \wedge s < x\}$ $t \Longrightarrow r \in \mathbb{R} \Longrightarrow 0 < r \Longrightarrow x \leq t + r$
for x r t :: *hypreal*
⟨*proof*⟩

lemma *hypreal-setle-less-trans*: $S *\!<= x \Longrightarrow x < y \Longrightarrow S *\!<= y$
for x y :: *hypreal*
⟨*proof*⟩

lemma *hypreal-gt-isUb*: *isUb* R S $x \Longrightarrow x < y \Longrightarrow y \in R \Longrightarrow$ *isUb* R S y
for x y :: *hypreal*
⟨*proof*⟩

lemma *lemma-st-part-gt-ub*: $x \in$ *HFinite* $\Longrightarrow x < y \Longrightarrow y \in \mathbb{R} \Longrightarrow$ *isUb* $\mathbb{R}$ $\{s.\ s \in \mathbb{R} \wedge s < x\}$ y
for x y :: *hypreal*

⟨*proof*⟩

lemma *lemma-minus-le-zero*: $t \le t + -r \Longrightarrow r \le 0$
for $r\ t$ :: *hypreal*
⟨*proof*⟩

lemma *lemma-st-part-le2*:
$x \in \mathit{HFinite} \Longrightarrow \mathit{isLub}\ \mathbb{R}\ \{s.\ s \in \mathbb{R} \wedge s < x\}\ t \Longrightarrow r \in \mathbb{R} \Longrightarrow 0 < r \Longrightarrow t + -r \le x$
for $x\ r\ t$ :: *hypreal*
⟨*proof*⟩

lemma *lemma-st-part1a*:
$x \in \mathit{HFinite} \Longrightarrow \mathit{isLub}\ \mathbb{R}\ \{s.\ s \in \mathbb{R} \wedge s < x\}\ t \Longrightarrow r \in \mathbb{R} \Longrightarrow 0 < r \Longrightarrow x + -t \le r$
for $x\ r\ t$ :: *hypreal*
⟨*proof*⟩

lemma *lemma-st-part2a*:
$x \in \mathit{HFinite} \Longrightarrow \mathit{isLub}\ \mathbb{R}\ \{s.\ s \in \mathbb{R} \wedge s < x\}\ t \Longrightarrow r \in \mathbb{R} \Longrightarrow 0 < r \Longrightarrow -\ (x + -t) \le r$
for $x\ r\ t$ :: *hypreal*
⟨*proof*⟩

lemma *lemma-SReal-ub*: $x \in \mathbb{R} \Longrightarrow \mathit{isUb}\ \mathbb{R}\ \{s.\ s \in \mathbb{R} \wedge s < x\}\ x$
for x :: *hypreal*
⟨*proof*⟩

lemma *lemma-SReal-lub*: $x \in \mathbb{R} \Longrightarrow \mathit{isLub}\ \mathbb{R}\ \{s.\ s \in \mathbb{R} \wedge s < x\}\ x$
for x :: *hypreal*
⟨*proof*⟩

lemma *lemma-st-part-not-eq1*:
$x \in \mathit{HFinite} \Longrightarrow \mathit{isLub}\ \mathbb{R}\ \{s.\ s \in \mathbb{R} \wedge s < x\}\ t \Longrightarrow r \in \mathbb{R} \Longrightarrow 0 < r \Longrightarrow x + -\ t \ne r$
for $x\ r\ t$ :: *hypreal*
⟨*proof*⟩

lemma *lemma-st-part-not-eq2*:
$x \in \mathit{HFinite} \Longrightarrow \mathit{isLub}\ \mathbb{R}\ \{s.\ s \in \mathbb{R} \wedge s < x\}\ t \Longrightarrow r \in \mathbb{R} \Longrightarrow 0 < r \Longrightarrow -\ (x + -t) \ne r$
for $x\ r\ t$ :: *hypreal*
⟨*proof*⟩

lemma *lemma-st-part-major*:
$x \in \mathit{HFinite} \Longrightarrow \mathit{isLub}\ \mathbb{R}\ \{s.\ s \in \mathbb{R} \wedge s < x\}\ t \Longrightarrow r \in \mathbb{R} \Longrightarrow 0 < r \Longrightarrow |x - t| < r$
for $x\ r\ t$ :: *hypreal*
⟨*proof*⟩

lemma *lemma-st-part-major2*:
$x \in HFinite \Longrightarrow isLub\ \mathbb{R}\ \{s.\ s \in \mathbb{R} \wedge s < x\}\ t \Longrightarrow \forall r \in Reals.\ 0 < r \longrightarrow |x - t| < r$
for $x\ t$ :: *hypreal*
⟨*proof*⟩

Existence of real and Standard Part Theorem.

lemma *lemma-st-part-Ex*: $x \in HFinite \Longrightarrow \exists t{\in}Reals.\ \forall r \in Reals.\ 0 < r \longrightarrow |x - t| < r$
for x :: *hypreal*
⟨*proof*⟩

lemma *st-part-Ex*: $x \in HFinite \Longrightarrow \exists t{\in}Reals.\ x \approx t$
for x :: *hypreal*
⟨*proof*⟩

There is a unique real infinitely close.

lemma *st-part-Ex1*: $x \in HFinite \Longrightarrow \exists!t{::}hypreal.\ t \in \mathbb{R} \wedge x \approx t$
⟨*proof*⟩

6.3 Finite, Infinite and Infinitesimal

lemma *HFinite-Int-HInfinite-empty* [*simp*]: *HFinite Int HInfinite* = {}
⟨*proof*⟩

lemma *HFinite-not-HInfinite*:
assumes x: $x \in HFinite$
shows $x \notin HInfinite$
⟨*proof*⟩

lemma *not-HFinite-HInfinite*: $x \notin HFinite \Longrightarrow x \in HInfinite$
⟨*proof*⟩

lemma *HInfinite-HFinite-disj*: $x \in HInfinite \vee x \in HFinite$
⟨*proof*⟩

lemma *HInfinite-HFinite-iff*: $x \in HInfinite \longleftrightarrow x \notin HFinite$
⟨*proof*⟩

lemma *HFinite-HInfinite-iff*: $x \in HFinite \longleftrightarrow x \notin HInfinite$
⟨*proof*⟩

lemma *HInfinite-diff-HFinite-Infinitesimal-disj*:
$x \notin Infinitesimal \Longrightarrow x \in HInfinite \vee x \in HFinite - Infinitesimal$
⟨*proof*⟩

lemma *HFinite-inverse*: $x \in HFinite \Longrightarrow x \notin Infinitesimal \Longrightarrow inverse\ x \in HFinite$
for x :: $'a$::*real-normed-div-algebra star*
⟨*proof*⟩

lemma *HFinite-inverse2*: $x \in HFinite - Infinitesimal \Longrightarrow inverse\ x \in HFinite$
for x :: $'a$::*real-normed-div-algebra star*
⟨*proof*⟩

Stronger statement possible in fact.

lemma *Infinitesimal-inverse-HFinite*: $x \notin Infinitesimal \Longrightarrow inverse\ x \in HFinite$
for x :: $'a$::*real-normed-div-algebra star*
⟨*proof*⟩

lemma *HFinite-not-Infinitesimal-inverse*:
$x \in HFinite - Infinitesimal \Longrightarrow inverse\ x \in HFinite - Infinitesimal$
for x :: $'a$::*real-normed-div-algebra star*
⟨*proof*⟩

lemma *approx-inverse*: $x \approx y \Longrightarrow y \in HFinite - Infinitesimal \Longrightarrow inverse\ x \approx inverse\ y$
for $x\ y$ :: $'a$::*real-normed-div-algebra star*
⟨*proof*⟩

lemmas *star-of-approx-inverse* = *star-of-HFinite-diff-Infinitesimal* [*THEN* [*2*] *approx-inverse*]
lemmas *hypreal-of-real-approx-inverse* = *hypreal-of-real-HFinite-diff-Infinitesimal* [*THEN* [*2*] *approx-inverse*]

lemma *inverse-add-Infinitesimal-approx*:
$x \in HFinite - Infinitesimal \Longrightarrow h \in Infinitesimal \Longrightarrow inverse\ (x + h) \approx inverse\ x$
for $x\ h$ :: $'a$::*real-normed-div-algebra star*
⟨*proof*⟩

lemma *inverse-add-Infinitesimal-approx2*:
$x \in HFinite - Infinitesimal \Longrightarrow h \in Infinitesimal \Longrightarrow inverse\ (h + x) \approx inverse\ x$
for $x\ h$ :: $'a$::*real-normed-div-algebra star*
⟨*proof*⟩

lemma *inverse-add-Infinitesimal-approx-Infinitesimal*:
$x \in HFinite - Infinitesimal \Longrightarrow h \in Infinitesimal \Longrightarrow inverse\ (x + h) - inverse\ x \approx h$
for $x\ h$ :: $'a$::*real-normed-div-algebra star*
⟨*proof*⟩

lemma *Infinitesimal-square-iff*: $x \in Infinitesimal \longleftrightarrow x * x \in Infinitesimal$
for x :: $'a$::*real-normed-div-algebra star*

⟨*proof*⟩
declare *Infinitesimal-square-iff* [*symmetric*, *simp*]

lemma *HFinite-square-iff* [*simp*]: $x * x \in \mathit{HFinite} \longleftrightarrow x \in \mathit{HFinite}$
for x :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *HInfinite-square-iff* [*simp*]: $x * x \in \mathit{HInfinite} \longleftrightarrow x \in \mathit{HInfinite}$
for x :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *approx-HFinite-mult-cancel*: $a \in \mathit{HFinite} - \mathit{Infinitesimal} \Longrightarrow a * w \approx a * z \Longrightarrow w \approx z$
for $a\ w\ z$ :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *approx-HFinite-mult-cancel-iff1*: $a \in \mathit{HFinite} - \mathit{Infinitesimal} \Longrightarrow a * w \approx a * z \longleftrightarrow w \approx z$
for $a\ w\ z$ :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *HInfinite-HFinite-add-cancel*: $x + y \in \mathit{HInfinite} \Longrightarrow y \in \mathit{HFinite} \Longrightarrow x \in \mathit{HInfinite}$
⟨*proof*⟩

lemma *HInfinite-HFinite-add*: $x \in \mathit{HInfinite} \Longrightarrow y \in \mathit{HFinite} \Longrightarrow x + y \in \mathit{HInfinite}$
⟨*proof*⟩

lemma *HInfinite-ge-HInfinite*: $x \in \mathit{HInfinite} \Longrightarrow x \le y \Longrightarrow 0 \le x \Longrightarrow y \in \mathit{HInfinite}$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *Infinitesimal-inverse-HInfinite*: $x \in \mathit{Infinitesimal} \Longrightarrow x \ne 0 \Longrightarrow \mathit{inverse}\ x \in \mathit{HInfinite}$
for x :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *HInfinite-HFinite-not-Infinitesimal-mult*:
$x \in \mathit{HInfinite} \Longrightarrow y \in \mathit{HFinite} - \mathit{Infinitesimal} \Longrightarrow x * y \in \mathit{HInfinite}$
for $x\ y$ :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *HInfinite-HFinite-not-Infinitesimal-mult2*:
$x \in \mathit{HInfinite} \Longrightarrow y \in \mathit{HFinite} - \mathit{Infinitesimal} \Longrightarrow y * x \in \mathit{HInfinite}$
for $x\ y$ :: *'a::real-normed-div-algebra star*
⟨*proof*⟩

lemma *HInfinite-gt-SReal*: $x \in \mathit{HInfinite} \Longrightarrow 0 < x \Longrightarrow y \in \mathbb{R} \Longrightarrow y < x$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *HInfinite-gt-zero-gt-one*: $x \in \mathit{HInfinite} \Longrightarrow 0 < x \Longrightarrow 1 < x$
for x :: *hypreal*
⟨*proof*⟩

lemma *not-HInfinite-one* [*simp*]: $1 \notin \mathit{HInfinite}$
⟨*proof*⟩

lemma *approx-hrabs-disj*: $|x| \approx x \vee |x| \approx -x$
for x :: *hypreal*
⟨*proof*⟩

6.4 Theorems about Monads

lemma *monad-hrabs-Un-subset*: $\mathit{monad}\ |x| \leq \mathit{monad}\ x \cup \mathit{monad}\ (-\ x)$
for x :: *hypreal*
⟨*proof*⟩

lemma *Infinitesimal-monad-eq*: $e \in \mathit{Infinitesimal} \Longrightarrow \mathit{monad}\ (x + e) = \mathit{monad}\ x$
⟨*proof*⟩

lemma *mem-monad-iff*: $u \in \mathit{monad}\ x \longleftrightarrow -\ u \in \mathit{monad}\ (-\ x)$
⟨*proof*⟩

lemma *Infinitesimal-monad-zero-iff*: $x \in \mathit{Infinitesimal} \longleftrightarrow x \in \mathit{monad}\ 0$
⟨*proof*⟩

lemma *monad-zero-minus-iff*: $x \in \mathit{monad}\ 0 \longleftrightarrow -\ x \in \mathit{monad}\ 0$
⟨*proof*⟩

lemma *monad-zero-hrabs-iff*: $x \in \mathit{monad}\ 0 \longleftrightarrow |x| \in \mathit{monad}\ 0$
for x :: *hypreal*
⟨*proof*⟩

lemma *mem-monad-self* [*simp*]: $x \in \mathit{monad}\ x$
⟨*proof*⟩

6.5 Proof that $x \approx y$ implies $|x| \approx |y|$

lemma *approx-subset-monad*: $x \approx y \Longrightarrow \{x, y\} \leq \mathit{monad}\ x$
⟨*proof*⟩

lemma *approx-subset-monad2*: $x \approx y \Longrightarrow \{x, y\} \leq \mathit{monad}\ y$
⟨*proof*⟩

lemma *mem-monad-approx*: $u \in \mathit{monad}\ x \Longrightarrow x \approx u$

⟨*proof*⟩

lemma *approx-mem-monad*: $x \approx u \Longrightarrow u \in monad\ x$
⟨*proof*⟩

lemma *approx-mem-monad2*: $x \approx u \Longrightarrow x \in monad\ u$
⟨*proof*⟩

lemma *approx-mem-monad-zero*: $x \approx y \Longrightarrow x \in monad\ 0 \Longrightarrow y \in monad\ 0$
⟨*proof*⟩

lemma *Infinitesimal-approx-hrabs*: $x \approx y \Longrightarrow x \in Infinitesimal \Longrightarrow |x| \approx |y|$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *less-Infinitesimal-less*: $0 < x \Longrightarrow x \notin Infinitesimal \Longrightarrow e \in Infinitesimal \Longrightarrow e < x$
for x :: *hypreal*
⟨*proof*⟩

lemma *Ball-mem-monad-gt-zero*: $0 < x \Longrightarrow x \notin Infinitesimal \Longrightarrow u \in monad\ x \Longrightarrow 0 < u$
for $u\ x$ :: *hypreal*
⟨*proof*⟩

lemma *Ball-mem-monad-less-zero*: $x < 0 \Longrightarrow x \notin Infinitesimal \Longrightarrow u \in monad\ x \Longrightarrow u < 0$
for $u\ x$ :: *hypreal*
⟨*proof*⟩

lemma *lemma-approx-gt-zero*: $0 < x \Longrightarrow x \notin Infinitesimal \Longrightarrow x \approx y \Longrightarrow 0 < y$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *lemma-approx-less-zero*: $x < 0 \Longrightarrow x \notin Infinitesimal \Longrightarrow x \approx y \Longrightarrow y < 0$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *approx-hrabs*: $x \approx y \Longrightarrow |x| \approx |y|$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *approx-hrabs-zero-cancel*: $|x| \approx 0 \Longrightarrow x \approx 0$
for x :: *hypreal*
⟨*proof*⟩

lemma *approx-hrabs-add-Infinitesimal*: $e \in Infinitesimal \Longrightarrow |x| \approx |x + e|$
for $e\ x$ :: *hypreal*

⟨*proof*⟩

lemma *approx-hrabs-add-minus-Infinitesimal*: $e \in \mathit{Infinitesimal}$ ==> $|x| \approx |x + -e|$
for $e\ x$ :: *hypreal*
⟨*proof*⟩

lemma *hrabs-add-Infinitesimal-cancel*:
$e \in \mathit{Infinitesimal} \Longrightarrow e' \in \mathit{Infinitesimal} \Longrightarrow |x + e| = |y + e'| \Longrightarrow |x| \approx |y|$
for $e\ e'\ x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *hrabs-add-minus-Infinitesimal-cancel*:
$e \in \mathit{Infinitesimal} \Longrightarrow e' \in \mathit{Infinitesimal} \Longrightarrow |x + -e| = |y + -e'| \Longrightarrow |x| \approx |y|$
for $e\ e'\ x\ y$ :: *hypreal*
⟨*proof*⟩

6.6 More *HFinite* and *Infinitesimal* Theorems

Interesting slightly counterintuitive theorem: necessary for proving that an open interval is an NS open set.

lemma *Infinitesimal-add-hypreal-of-real-less*:
$x < y \Longrightarrow u \in \mathit{Infinitesimal} \Longrightarrow \mathit{hypreal\text{-}of\text{-}real}\ x + u < \mathit{hypreal\text{-}of\text{-}real}\ y$
⟨*proof*⟩

lemma *Infinitesimal-add-hrabs-hypreal-of-real-less*:
$x \in \mathit{Infinitesimal} \Longrightarrow |\mathit{hypreal\text{-}of\text{-}real}\ r| < \mathit{hypreal\text{-}of\text{-}real}\ y \Longrightarrow$
$|\mathit{hypreal\text{-}of\text{-}real}\ r + x| < \mathit{hypreal\text{-}of\text{-}real}\ y$
⟨*proof*⟩

lemma *Infinitesimal-add-hrabs-hypreal-of-real-less2*:
$x \in \mathit{Infinitesimal} \Longrightarrow |\mathit{hypreal\text{-}of\text{-}real}\ r| < \mathit{hypreal\text{-}of\text{-}real}\ y \Longrightarrow$
$|x + \mathit{hypreal\text{-}of\text{-}real}\ r| < \mathit{hypreal\text{-}of\text{-}real}\ y$
⟨*proof*⟩

lemma *hypreal-of-real-le-add-Infininitesimal-cancel*:
$u \in \mathit{Infinitesimal} \Longrightarrow v \in \mathit{Infinitesimal} \Longrightarrow$
$\mathit{hypreal\text{-}of\text{-}real}\ x + u \le \mathit{hypreal\text{-}of\text{-}real}\ y + v \Longrightarrow$
$\mathit{hypreal\text{-}of\text{-}real}\ x \le \mathit{hypreal\text{-}of\text{-}real}\ y$
⟨*proof*⟩

lemma *hypreal-of-real-le-add-Infininitesimal-cancel2*:
$u \in \mathit{Infinitesimal} \Longrightarrow v \in \mathit{Infinitesimal} \Longrightarrow$
$\mathit{hypreal\text{-}of\text{-}real}\ x + u \le \mathit{hypreal\text{-}of\text{-}real}\ y + v \Longrightarrow x \le y$
⟨*proof*⟩

lemma *hypreal-of-real-less-Infinitesimal-le-zero*:
$\mathit{hypreal\text{-}of\text{-}real}\ x < e \Longrightarrow e \in \mathit{Infinitesimal} \Longrightarrow \mathit{hypreal\text{-}of\text{-}real}\ x \le \mathit{0}$
⟨*proof*⟩

lemma *Infinitesimal-add-not-zero*: $h \in \mathit{Infinitesimal} \Longrightarrow x \neq 0 \Longrightarrow \mathit{star\text{-}of}\ x + h \neq 0$
⟨*proof*⟩

lemma *Infinitesimal-square-cancel* [*simp*]: $x * x + y * y \in \mathit{Infinitesimal} \Longrightarrow x * x \in \mathit{Infinitesimal}$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *HFinite-square-cancel* [*simp*]: $x * x + y * y \in \mathit{HFinite} \Longrightarrow x * x \in \mathit{HFinite}$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *Infinitesimal-square-cancel2* [*simp*]: $x * x + y * y \in \mathit{Infinitesimal} \Longrightarrow y * y \in \mathit{Infinitesimal}$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *HFinite-square-cancel2* [*simp*]: $x * x + y * y \in \mathit{HFinite} \Longrightarrow y * y \in \mathit{HFinite}$
for $x\ y$ :: *hypreal*
⟨*proof*⟩

lemma *Infinitesimal-sum-square-cancel* [*simp*]:
$x * x + y * y + z * z \in \mathit{Infinitesimal} \Longrightarrow x * x \in \mathit{Infinitesimal}$
for $x\ y\ z$ :: *hypreal*
⟨*proof*⟩

lemma *HFinite-sum-square-cancel* [*simp*]: $x * x + y * y + z * z \in \mathit{HFinite} \Longrightarrow x * x \in \mathit{HFinite}$
for $x\ y\ z$ :: *hypreal*
⟨*proof*⟩

lemma *Infinitesimal-sum-square-cancel2* [*simp*]:
$y * y + x * x + z * z \in \mathit{Infinitesimal} \Longrightarrow x * x \in \mathit{Infinitesimal}$
for $x\ y\ z$ :: *hypreal*
⟨*proof*⟩

lemma *HFinite-sum-square-cancel2* [*simp*]: $y * y + x * x + z * z \in \mathit{HFinite} \Longrightarrow x * x \in \mathit{HFinite}$
for $x\ y\ z$ :: *hypreal*
⟨*proof*⟩

lemma *Infinitesimal-sum-square-cancel3* [*simp*]:
$z * z + y * y + x * x \in \mathit{Infinitesimal} \Longrightarrow x * x \in \mathit{Infinitesimal}$
for $x\ y\ z$ :: *hypreal*
⟨*proof*⟩

lemma *HFinite-sum-square-cancel3* [*simp*]: $z * z + y * y + x * x \in HFinite \Longrightarrow x * x \in HFinite$
for $x\ y\ z$ :: *hypreal*
⟨*proof*⟩

lemma *monad-hrabs-less*: $y \in monad\ x \Longrightarrow 0 < hypreal\text{-}of\text{-}real\ e \Longrightarrow |y - x| < hypreal\text{-}of\text{-}real\ e$
⟨*proof*⟩

lemma *mem-monad-SReal-HFinite*: $x \in monad\ (hypreal\text{-}of\text{-}real\ \ a) \Longrightarrow x \in HFinite$
⟨*proof*⟩

6.7 Theorems about Standard Part

lemma *st-approx-self*: $x \in HFinite \Longrightarrow st\ x \approx x$
⟨*proof*⟩

lemma *st-SReal*: $x \in HFinite \Longrightarrow st\ x \in \mathbb{R}$
⟨*proof*⟩

lemma *st-HFinite*: $x \in HFinite \Longrightarrow st\ x \in HFinite$
⟨*proof*⟩

lemma *st-unique*: $r \in \mathbb{R} \Longrightarrow r \approx x \Longrightarrow st\ x = r$
⟨*proof*⟩

lemma *st-SReal-eq*: $x \in \mathbb{R} \Longrightarrow st\ x = x$
⟨*proof*⟩

lemma *st-hypreal-of-real* [*simp*]: $st\ (hypreal\text{-}of\text{-}real\ x) = hypreal\text{-}of\text{-}real\ x$
⟨*proof*⟩

lemma *st-eq-approx*: $x \in HFinite \Longrightarrow y \in HFinite \Longrightarrow st\ x = st\ y \Longrightarrow x \approx y$
⟨*proof*⟩

lemma *approx-st-eq*:
assumes x: $x \in HFinite$ **and** y: $y \in HFinite$ **and** xy: $x \approx y$
shows $st\ x = st\ y$
⟨*proof*⟩

lemma *st-eq-approx-iff*: $x \in HFinite \Longrightarrow y \in HFinite \Longrightarrow x \approx y \longleftrightarrow st\ x = st\ y$
⟨*proof*⟩

lemma *st-Infinitesimal-add-SReal*: $x \in \mathbb{R} \Longrightarrow e \in Infinitesimal \Longrightarrow st\ (x + e) = x$
⟨*proof*⟩

lemma *st-Infinitesimal-add-SReal2*: $x \in \mathbb{R} \Longrightarrow e \in Infinitesimal \Longrightarrow st\ (e + x)$

$= x$
⟨*proof*⟩

lemma *HFinite-st-Infinitesimal-add*: $x \in HFinite \Longrightarrow \exists\, e \in Infinitesimal.\ x = st(x) + e$
⟨*proof*⟩

lemma *st-add*: $x \in HFinite \Longrightarrow y \in HFinite \Longrightarrow st\ (x + y) = st\ x + st\ y$
⟨*proof*⟩

lemma *st-numeral* [*simp*]: $st\ (numeral\ w) = numeral\ w$
⟨*proof*⟩

lemma *st-neg-numeral* [*simp*]: $st\ (-\ numeral\ w) = -\ numeral\ w$
⟨*proof*⟩

lemma *st-0* [*simp*]: $st\ 0 = 0$
⟨*proof*⟩

lemma *st-1* [*simp*]: $st\ 1 = 1$
⟨*proof*⟩

lemma *st-neg-1* [*simp*]: $st\ (-\ 1) = -\ 1$
⟨*proof*⟩

lemma *st-minus*: $x \in HFinite \Longrightarrow st\ (-\ x) = -\ st\ x$
⟨*proof*⟩

lemma *st-diff*: $[\![x \in HFinite;\ y \in HFinite]\!] \Longrightarrow st\ (x - y) = st\ x - st\ y$
⟨*proof*⟩

lemma *st-mult*: $[\![x \in HFinite;\ y \in HFinite]\!] \Longrightarrow st\ (x * y) = st\ x * st\ y$
⟨*proof*⟩

lemma *st-Infinitesimal*: $x \in Infinitesimal \Longrightarrow st\ x = 0$
⟨*proof*⟩

lemma *st-not-Infinitesimal*: $st(x) \neq 0 \Longrightarrow x \notin Infinitesimal$
⟨*proof*⟩

lemma *st-inverse*: $x \in HFinite \Longrightarrow st\ x \neq 0 \Longrightarrow st\ (inverse\ x) = inverse\ (st\ x)$
⟨*proof*⟩

lemma *st-divide* [*simp*]: $x \in HFinite \Longrightarrow y \in HFinite \Longrightarrow st\ y \neq 0 \Longrightarrow st\ (x\ /\ y) = st\ x\ /\ st\ y$
⟨*proof*⟩

lemma *st-idempotent* [*simp*]: $x \in HFinite \Longrightarrow st\ (st\ x) = st\ x$
⟨*proof*⟩

lemma *Infinitesimal-add-st-less*:
$x \in HFinite \Longrightarrow y \in HFinite \Longrightarrow u \in Infinitesimal \Longrightarrow st\ x < st\ y \Longrightarrow st\ x + u < st\ y$
⟨*proof*⟩

lemma *Infinitesimal-add-st-le-cancel*:
$x \in HFinite \Longrightarrow y \in HFinite \Longrightarrow u \in Infinitesimal \Longrightarrow$
$st\ x \le st\ y + u \Longrightarrow st\ x \le st\ y$
⟨*proof*⟩

lemma *st-le*: $x \in HFinite \Longrightarrow y \in HFinite \Longrightarrow x \le y \Longrightarrow st\ x \le st\ y$
⟨*proof*⟩

lemma *st-zero-le*: $0 \le x \Longrightarrow x \in HFinite \Longrightarrow 0 \le st\ x$
⟨*proof*⟩

lemma *st-zero-ge*: $x \le 0 \Longrightarrow x \in HFinite \Longrightarrow st\ x \le 0$
⟨*proof*⟩

lemma *st-hrabs*: $x \in HFinite \Longrightarrow |st\ x| = st\ |x|$
⟨*proof*⟩

6.8 Alternative Definitions using Free Ultrafilter

6.8.1 *HFinite*

lemma *HFinite-FreeUltrafilterNat*:
$star\text{-}n\ X \in HFinite \Longrightarrow \exists u.\ eventually\ (\lambda n.\ norm\ (X\ n) < u)\ \mathcal{U}$
⟨*proof*⟩

lemma *FreeUltrafilterNat-HFinite*:
$\exists u.\ eventually\ (\lambda n.\ norm\ (X\ n) < u)\ \mathcal{U} \Longrightarrow star\text{-}n\ X \in HFinite$
⟨*proof*⟩

lemma *HFinite-FreeUltrafilterNat-iff*:
$star\text{-}n\ X \in HFinite \longleftrightarrow (\exists u.\ eventually\ (\lambda n.\ norm\ (X\ n) < u)\ \mathcal{U})$
⟨*proof*⟩

6.8.2 *HInfinite*

lemma *lemma-Compl-eq*: $-\ \{n.\ u < norm\ (f\ n)\} = \{n.\ norm\ (f\ n) \le u\}$
⟨*proof*⟩

lemma *lemma-Compl-eq2*: $-\ \{n.\ norm\ (f\ n) < u\} = \{n.\ u \le norm\ (f\ n)\}$
⟨*proof*⟩

lemma *lemma-Int-eq1*: $\{n.\ norm\ (f\ n) \le u\}\ Int\ \{n.\ u \le norm\ (f\ n)\} = \{n.\ norm(f\ n) = u\}$
⟨*proof*⟩

lemma *lemma-FreeUltrafilterNat-one*: $\{n.\ norm\ (f\ n) = u\} \le \{n.\ norm\ (f\ n) < u + (1{::}real)\}$
⟨*proof*⟩

Exclude this type of sets from free ultrafilter for Infinite numbers!

lemma *FreeUltrafilterNat-const-Finite*:
$eventually\ (\lambda n.\ norm\ (X\ n) = u)\ \mathcal{U} \Longrightarrow star\text{-}n\ X \in HFinite$
⟨*proof*⟩

lemma *HInfinite-FreeUltrafilterNat*:
$star\text{-}n\ X \in HInfinite \Longrightarrow \forall u.\ eventually\ (\lambda n.\ u < norm\ (X\ n))\ \mathcal{U}$
⟨*proof*⟩

lemma *lemma-Int-HI*: $\{n.\ norm\ (Xa\ n) < u\} \cap \{n.\ X\ n = Xa\ n\} \subseteq \{n.\ norm\ (X\ n) < u\}$
for $u :: real$
⟨*proof*⟩

lemma *lemma-Int-HIa*: $\{n.\ u < norm\ (X\ n)\} \cap \{n.\ norm\ (X\ n) < u\} = \{\}$
⟨*proof*⟩

lemma *FreeUltrafilterNat-HInfinite*:
$\forall u.\ eventually\ (\lambda n.\ u < norm\ (X\ n))\ \mathcal{U} \Longrightarrow star\text{-}n\ X \in HInfinite$
⟨*proof*⟩

lemma *HInfinite-FreeUltrafilterNat-iff*:
$star\text{-}n\ X \in HInfinite \longleftrightarrow (\forall u.\ eventually\ (\lambda n.\ u < norm\ (X\ n))\ \mathcal{U})$
⟨*proof*⟩

6.8.3 *Infinitesimal*

lemma *ball-SReal-eq*: $(\forall x{::}hypreal \in Reals.\ P\ x) \longleftrightarrow (\forall x{::}real.\ P\ (star\text{-}of\ x))$
⟨*proof*⟩

lemma *Infinitesimal-FreeUltrafilterNat*:
$star\text{-}n\ X \in Infinitesimal \Longrightarrow \forall u{>}0.\ eventually\ (\lambda n.\ norm\ (X\ n) < u)\ \mathcal{U}$
⟨*proof*⟩

lemma *FreeUltrafilterNat-Infinitesimal*:
$\forall u{>}0.\ eventually\ (\lambda n.\ norm\ (X\ n) < u)\ \mathcal{U} \Longrightarrow star\text{-}n\ X \in Infinitesimal$
⟨*proof*⟩

lemma *Infinitesimal-FreeUltrafilterNat-iff*:
$(star\text{-}n\ X \in Infinitesimal) = (\forall u{>}0.\ eventually\ (\lambda n.\ norm\ (X\ n) < u)\ \mathcal{U})$
⟨*proof*⟩

Infinitesimals as smaller than $1/n$ for all $n{::}nat$ (> 0).

lemma *lemma-Infinitesimal*: $(\forall r.\ 0 < r \longrightarrow x < r) \longleftrightarrow (\forall n.\ x < inverse\ (real$

$(Suc\ n)))$
⟨proof⟩

lemma *lemma-Infinitesimal2*:
$(\forall r \in Reals.\ 0 < r \longrightarrow x < r) \longleftrightarrow (\forall n.\ x < inverse(hypreal\text{-}of\text{-}nat\ (Suc\ n)))$
⟨proof⟩

lemma *Infinitesimal-hypreal-of-nat-iff*:
$Infinitesimal = \{x.\ \forall n.\ hnorm\ x < inverse\ (hypreal\text{-}of\text{-}nat\ (Suc\ n))\}$
⟨proof⟩

6.9 Proof that ω is an infinite number

It will follow that ε is an infinitesimal number.

lemma *Suc-Un-eq*: $\{n.\ n < Suc\ m\} = \{n.\ n < m\}\ Un\ \{n.\ n = m\}$
⟨proof⟩

Prove that any segment is finite and hence cannot belong to $\mathcal{U}$.

lemma *finite-real-of-nat-segment*: $finite\ \{n::nat.\ real\ n < real\ (m::nat)\}$
⟨proof⟩

lemma *finite-real-of-nat-less-real*: $finite\ \{n::nat.\ real\ n < u\}$
⟨proof⟩

lemma *lemma-real-le-Un-eq*: $\{n.\ f\ n \leq u\} = \{n.\ f\ n < u\} \cup \{n.\ u = (f\ n :: real)\}$
⟨proof⟩

lemma *finite-real-of-nat-le-real*: $finite\ \{n::nat.\ real\ n \leq u\}$
⟨proof⟩

lemma *finite-rabs-real-of-nat-le-real*: $finite\ \{n::nat.\ |real\ n| \leq u\}$
⟨proof⟩

lemma *rabs-real-of-nat-le-real-FreeUltrafilterNat*:
$\neg\ eventually\ (\lambda n.\ |real\ n| \leq u)\ \mathcal{U}$
⟨proof⟩

lemma *FreeUltrafilterNat-nat-gt-real*: $eventually\ (\lambda n.\ u < real\ n)\ \mathcal{U}$
⟨proof⟩

The complement of $\{n.\ |real\ n| \leq u\} = \{n.\ u < |real\ n|\}$ is in $\mathcal{U}$ by property of (free) ultrafilters.

lemma *Compl-real-le-eq*: $-\ \{n::nat.\ real\ n \leq u\} = \{n.\ u < real\ n\}$
⟨proof⟩

ω is a member of *HInfinite*.

theorem *HInfinite-omega* [*simp*]: $\omega \in HInfinite$

⟨*proof*⟩

Epsilon is a member of Infinitesimal.

lemma *Infinitesimal-epsilon* [*simp*]: $\varepsilon \in$ *Infinitesimal*
⟨*proof*⟩

lemma *HFinite-epsilon* [*simp*]: $\varepsilon \in$ *HFinite*
⟨*proof*⟩

lemma *epsilon-approx-zero* [*simp*]: $\varepsilon \approx 0$
⟨*proof*⟩

Needed for proof that we define a hyperreal [<$X(n)$] $\approx$ *hypreal-of-real a* given that $\forall n.\ |X\ n\ -\ a| < 1/n$. Used in proof of *NSLIM* $\Rightarrow$ *LIM*.

lemma *real-of-nat-less-inverse-iff*: $0 < u \Longrightarrow u <$ *inverse* (*real*(*Suc n*)) $\longleftrightarrow$ *real*(*Suc n*) $<$ *inverse u*
⟨*proof*⟩

lemma *finite-inverse-real-of-posnat-gt-real*: $0 < u \Longrightarrow$ *finite* {*n*. $u <$ *inverse* (*real* (*Suc n*))}
⟨*proof*⟩

lemma *lemma-real-le-Un-eq2*:
{*n*. $u \leq$ *inverse*(*real*(*Suc n*))} =
{*n*. $u <$ *inverse*(*real*(*Suc n*))} $\cup$ {*n*. $u =$ *inverse*(*real*(*Suc n*))}
⟨*proof*⟩

lemma *finite-inverse-real-of-posnat-ge-real*: $0 < u \Longrightarrow$ *finite* {*n*. $u \leq$ *inverse* (*real* (*Suc n*))}
⟨*proof*⟩

lemma *inverse-real-of-posnat-ge-real-FreeUltrafilterNat*:
$0 < u \Longrightarrow \neg$ *eventually* (λn. $u \leq$ *inverse*(*real*(*Suc n*))) $\mathcal{U}$
⟨*proof*⟩

The complement of {*n*. $u \leq$ *inverse*(*real*(*Suc n*))} = {*n*. *inverse* (*real* (*Suc n*)) $< u$} is in $\mathcal{U}$ by property of (free) ultrafilters.

lemma *Compl-le-inverse-eq*: $-$ {*n*. $u \leq$ *inverse*(*real*(*Suc n*))} = {*n*. *inverse*(*real*(*Suc n*)) $< u$}
⟨*proof*⟩

lemma *FreeUltrafilterNat-inverse-real-of-posnat*:
$0 < u \Longrightarrow$ *eventually* (λn. *inverse*(*real*(*Suc n*)) $< u$) $\mathcal{U}$
⟨*proof*⟩

Example of an hypersequence (i.e. an extended standard sequence) whose term with an hypernatural suffix is an infinitesimal i.e. the whn'nth term of the hypersequence is a member of Infinitesimal

lemma *SEQ-Infinitesimal*: $(*f* (\lambda n::nat.\ inverse(real(Suc\ n))))\ whn \in Infinitesimal$
⟨*proof*⟩

Example where we get a hyperreal from a real sequence for which a particular property holds. The theorem is used in proofs about equivalence of nonstandard and standard neighbourhoods. Also used for equivalence of nonstandard ans standard definitions of pointwise limit.

$|X(n) - x| < 1/n \Longrightarrow [<X\ n>] - hypreal\text{-}of\text{-}real\ x| \in Infinitesimal$

lemma *real-seq-to-hypreal-Infinitesimal*:
$\forall n.\ norm\ (X\ n - x) < inverse\ (real\ (Suc\ n)) \Longrightarrow star\text{-}n\ X - star\text{-}of\ x \in Infinitesimal$
⟨*proof*⟩

lemma *real-seq-to-hypreal-approx*:
$\forall n.\ norm\ (X\ n - x) < inverse\ (real\ (Suc\ n)) \Longrightarrow star\text{-}n\ X \approx star\text{-}of\ x$
⟨*proof*⟩

lemma *real-seq-to-hypreal-approx2*:
$\forall n.\ norm\ (x - X\ n) < inverse(real(Suc\ n)) \Longrightarrow star\text{-}n\ X \approx star\text{-}of\ x$
⟨*proof*⟩

lemma *real-seq-to-hypreal-Infinitesimal2*:
$\forall n.\ norm(X\ n - Y\ n) < inverse(real(Suc\ n)) \Longrightarrow star\text{-}n\ X - star\text{-}n\ Y \in Infinitesimal$
⟨*proof*⟩

end

7 Nonstandard Complex Numbers

theory *NSComplex*
imports *NSA*
begin

type-synonym *hcomplex = complex star*

abbreviation *hcomplex-of-complex* :: *complex ⇒ complex star*
where *hcomplex-of-complex ≡ star-of*

abbreviation *hcmod* :: *complex star ⇒ real star*
where *hcmod ≡ hnorm*

7.0.1 Real and Imaginary parts

definition *hRe* :: *hcomplex ⇒ hypreal*
where *hRe = *f* Re*

definition *hIm* :: *hcomplex ⇒ hypreal*

where *hIm* = *∗f∗ Im*

7.0.2 Imaginary unit

definition *iii* :: *hcomplex*
where *iii* = *star-of* i

7.0.3 Complex conjugate

definition *hcnj* :: *hcomplex* ⇒ *hcomplex*
where *hcnj* = *∗f∗ cnj*

7.0.4 Argand

definition *hsgn* :: *hcomplex* ⇒ *hcomplex*
where *hsgn* = *∗f∗ sgn*

definition *harg* :: *hcomplex* ⇒ *hypreal*
where *harg* = *∗f∗ arg*

definition — abbreviation for *cos a* + *i sin a*
hcis :: *hypreal* ⇒ *hcomplex*
where *hcis* = *∗f∗ cis*

7.0.5 Injection from hyperreals

abbreviation *hcomplex-of-hypreal* :: *hypreal* ⇒ *hcomplex*
where *hcomplex-of-hypreal* ≡ *of-hypreal*

definition — abbreviation for *r* ∗ (*cos a* + *i sin a*)
hrcis :: *hypreal* ⇒ *hypreal* ⇒ *hcomplex*
where *hrcis* = *∗f2∗ rcis*

7.0.6 e ˆ $(x + iy)$

definition *hExp* :: *hcomplex* ⇒ *hcomplex*
where *hExp* = *∗f∗ exp*

definition *HComplex* :: *hypreal* ⇒ *hypreal* ⇒ *hcomplex*
where *HComplex* = *∗f2∗ Complex*

lemmas *hcomplex-defs* [*transfer-unfold*] =
hRe-def hIm-def iii-def hcnj-def hsgn-def harg-def hcis-def
hrcis-def hExp-def HComplex-def

lemma *Standard-hRe* [*simp*]: *x* ∈ *Standard* ⟹ *hRe x* ∈ *Standard*
⟨*proof*⟩

lemma *Standard-hIm* [*simp*]: *x* ∈ *Standard* ⟹ *hIm x* ∈ *Standard*
⟨*proof*⟩

lemma *Standard-iii* [*simp*]: *iii* ∈ *Standard*
⟨*proof*⟩

lemma *Standard-hcnj* [*simp*]: *x* ∈ *Standard* ⟹ *hcnj x* ∈ *Standard*
⟨*proof*⟩

lemma *Standard-hsgn* [*simp*]: *x* ∈ *Standard* ⟹ *hsgn x* ∈ *Standard*
⟨*proof*⟩

lemma *Standard-harg* [*simp*]: *x* ∈ *Standard* ⟹ *harg x* ∈ *Standard*
⟨*proof*⟩

lemma *Standard-hcis* [*simp*]: *r* ∈ *Standard* ⟹ *hcis r* ∈ *Standard*
⟨*proof*⟩

lemma *Standard-hExp* [*simp*]: *x* ∈ *Standard* ⟹ *hExp x* ∈ *Standard*
⟨*proof*⟩

lemma *Standard-hrcis* [*simp*]: *r* ∈ *Standard* ⟹ *s* ∈ *Standard* ⟹ *hrcis r s* ∈ *Standard*
⟨*proof*⟩

lemma *Standard-HComplex* [*simp*]: *r* ∈ *Standard* ⟹ *s* ∈ *Standard* ⟹ *HComplex r s* ∈ *Standard*
⟨*proof*⟩

lemma *hcmod-def*: *hcmod* = ∗*f*∗ *cmod*
⟨*proof*⟩

7.1 Properties of Nonstandard Real and Imaginary Parts

lemma *hcomplex-hRe-hIm-cancel-iff*: ⋀*w z*. *w* = *z* ⟷ *hRe w* = *hRe z* ∧ *hIm w* = *hIm z*
⟨*proof*⟩

lemma *hcomplex-equality* [*intro?*]: ⋀*z w*. *hRe z* = *hRe w* ⟹ *hIm z* = *hIm w* ⟹ *z* = *w*
⟨*proof*⟩

lemma *hcomplex-hRe-zero* [*simp*]: *hRe 0* = *0*
⟨*proof*⟩

lemma *hcomplex-hIm-zero* [*simp*]: *hIm 0* = *0*
⟨*proof*⟩

lemma *hcomplex-hRe-one* [*simp*]: *hRe 1* = *1*
⟨*proof*⟩

lemma *hcomplex-hIm-one* [*simp*]: *hIm 1 = 0*
⟨*proof*⟩

7.2 Addition for Nonstandard Complex Numbers

lemma *hRe-add*: $\bigwedge x\ y.\ hRe\ (x + y) = hRe\ x + hRe\ y$
⟨*proof*⟩

lemma *hIm-add*: $\bigwedge x\ y.\ hIm\ (x + y) = hIm\ x + hIm\ y$
⟨*proof*⟩

7.3 More Minus Laws

lemma *hRe-minus*: $\bigwedge z.\ hRe\ (-\ z) = -\ hRe\ z$
⟨*proof*⟩

lemma *hIm-minus*: $\bigwedge z.\ hIm\ (-\ z) = -\ hIm\ z$
⟨*proof*⟩

lemma *hcomplex-add-minus-eq-minus*: $x + y = 0 \Longrightarrow x = -\ y$
for *x y* :: *hcomplex*
⟨*proof*⟩

lemma *hcomplex-i-mult-eq* [*simp*]: $iii * iii = -\ 1$
⟨*proof*⟩

lemma *hcomplex-i-mult-left* [*simp*]: $\bigwedge z.\ iii * (iii * z) = -\ z$
⟨*proof*⟩

lemma *hcomplex-i-not-zero* [*simp*]: $iii \neq 0$
⟨*proof*⟩

7.4 More Multiplication Laws

lemma *hcomplex-mult-minus-one*: $-\ 1 * z = -\ z$
for *z* :: *hcomplex*
⟨*proof*⟩

lemma *hcomplex-mult-minus-one-right*: $z * -\ 1 = -\ z$
for *z* :: *hcomplex*
⟨*proof*⟩

lemma *hcomplex-mult-left-cancel*: $c \neq 0 \Longrightarrow c * a = c * b \longleftrightarrow a = b$
for *a b c* :: *hcomplex*
⟨*proof*⟩

lemma *hcomplex-mult-right-cancel*: $c \neq 0 \Longrightarrow a * c = b * c \longleftrightarrow a = b$
for *a b c* :: *hcomplex*
⟨*proof*⟩

7.5 Subtraction and Division

lemma *hcomplex-diff-eq-eq* [*simp*]: $x - y = z \longleftrightarrow x = z + y$
for $x\ y\ z$ :: *hcomplex*
⟨*proof*⟩

7.6 Embedding Properties for *hcomplex-of-hypreal* Map

lemma *hRe-hcomplex-of-hypreal* [*simp*]: $\bigwedge z.\ hRe\ (hcomplex\text{-}of\text{-}hypreal\ z) = z$
⟨*proof*⟩

lemma *hIm-hcomplex-of-hypreal* [*simp*]: $\bigwedge z.\ hIm\ (hcomplex\text{-}of\text{-}hypreal\ z) = 0$
⟨*proof*⟩

lemma *hcomplex-of-hypreal-epsilon-not-zero* [*simp*]: $hcomplex\text{-}of\text{-}hypreal\ \varepsilon \neq 0$
⟨*proof*⟩

7.7 *HComplex* theorems

lemma *hRe-HComplex* [*simp*]: $\bigwedge x\ y.\ hRe\ (HComplex\ x\ y) = x$
⟨*proof*⟩

lemma *hIm-HComplex* [*simp*]: $\bigwedge x\ y.\ hIm\ (HComplex\ x\ y) = y$
⟨*proof*⟩

lemma *hcomplex-surj* [*simp*]: $\bigwedge z.\ HComplex\ (hRe\ z)\ (hIm\ z) = z$
⟨*proof*⟩

lemma *hcomplex-induct* [*case-names rect*]:
$(\bigwedge x\ y.\ P\ (HComplex\ x\ y)) \Longrightarrow P\ z$
⟨*proof*⟩

7.8 Modulus (Absolute Value) of Nonstandard Complex Number

lemma *hcomplex-of-hypreal-abs*:
$hcomplex\text{-}of\text{-}hypreal\ |x| = hcomplex\text{-}of\text{-}hypreal\ (hcmod\ (hcomplex\text{-}of\text{-}hypreal\ x))$
⟨*proof*⟩

lemma *HComplex-inject* [*simp*]: $\bigwedge x\ y\ x'\ y'.\ HComplex\ x\ y = HComplex\ x'\ y' \longleftrightarrow x = x' \land y = y'$
⟨*proof*⟩

lemma *HComplex-add* [*simp*]:
$\bigwedge x1\ y1\ x2\ y2.\ HComplex\ x1\ y1 + HComplex\ x2\ y2 = HComplex\ (x1 + x2)\ (y1 + y2)$
⟨*proof*⟩

lemma *HComplex-minus* [*simp*]: $\bigwedge x\ y.\ -\ HComplex\ x\ y = HComplex\ (-\ x)\ (-\ y)$

⟨*proof*⟩

lemma *HComplex-diff* [*simp*]:
$\bigwedge x1\ y1\ x2\ y2.\ HComplex\ x1\ y1 - HComplex\ x2\ y2 = HComplex\ (x1 - x2)\ (y1 - y2)$
⟨*proof*⟩

lemma *HComplex-mult* [*simp*]:
$\bigwedge x1\ y1\ x2\ y2.\ HComplex\ x1\ y1 * HComplex\ x2\ y2 = HComplex\ (x1*x2 - y1*y2)\ (x1*y2 + y1*x2)$
⟨*proof*⟩

HComplex-inverse is proved below.

lemma *hcomplex-of-hypreal-eq*: $\bigwedge r.\ hcomplex\text{-}of\text{-}hypreal\ r = HComplex\ r\ 0$
⟨*proof*⟩

lemma *HComplex-add-hcomplex-of-hypreal* [*simp*]:
$\bigwedge x\ y\ r.\ HComplex\ x\ y + hcomplex\text{-}of\text{-}hypreal\ r = HComplex\ (x + r)\ y$
⟨*proof*⟩

lemma *hcomplex-of-hypreal-add-HComplex* [*simp*]:
$\bigwedge r\ x\ y.\ hcomplex\text{-}of\text{-}hypreal\ r + HComplex\ x\ y = HComplex\ (r + x)\ y$
⟨*proof*⟩

lemma *HComplex-mult-hcomplex-of-hypreal*:
$\bigwedge x\ y\ r.\ HComplex\ x\ y * hcomplex\text{-}of\text{-}hypreal\ r = HComplex\ (x * r)\ (y * r)$
⟨*proof*⟩

lemma *hcomplex-of-hypreal-mult-HComplex*:
$\bigwedge r\ x\ y.\ hcomplex\text{-}of\text{-}hypreal\ r * HComplex\ x\ y = HComplex\ (r * x)\ (r * y)$
⟨*proof*⟩

lemma *i-hcomplex-of-hypreal* [*simp*]: $\bigwedge r.\ iii * hcomplex\text{-}of\text{-}hypreal\ r = HComplex\ 0\ r$
⟨*proof*⟩

lemma *hcomplex-of-hypreal-i* [*simp*]: $\bigwedge r.\ hcomplex\text{-}of\text{-}hypreal\ r * iii = HComplex\ 0\ r$
⟨*proof*⟩

7.9 Conjugation

lemma *hcomplex-hcnj-cancel-iff* [*iff*]: $\bigwedge x\ y.\ hcnj\ x = hcnj\ y \longleftrightarrow x = y$
⟨*proof*⟩

lemma *hcomplex-hcnj-hcnj* [*simp*]: $\bigwedge z.\ hcnj\ (hcnj\ z) = z$
⟨*proof*⟩

lemma *hcomplex-hcnj-hcomplex-of-hypreal* [*simp*]:

$\bigwedge x.$ *hcnj* (*hcomplex-of-hypreal x*) = *hcomplex-of-hypreal x*
⟨*proof*⟩

lemma *hcomplex-hmod-hcnj* [*simp*]: $\bigwedge z.$ *hcmod* (*hcnj z*) = *hcmod z*
⟨*proof*⟩

lemma *hcomplex-hcnj-minus*: $\bigwedge z.$ *hcnj* (− *z*) = − *hcnj z*
⟨*proof*⟩

lemma *hcomplex-hcnj-inverse*: $\bigwedge z.$ *hcnj* (*inverse z*) = *inverse* (*hcnj z*)
⟨*proof*⟩

lemma *hcomplex-hcnj-add*: $\bigwedge w\ z.$ *hcnj* (*w* + *z*) = *hcnj w* + *hcnj z*
⟨*proof*⟩

lemma *hcomplex-hcnj-diff*: $\bigwedge w\ z.$ *hcnj* (*w* − *z*) = *hcnj w* − *hcnj z*
⟨*proof*⟩

lemma *hcomplex-hcnj-mult*: $\bigwedge w\ z.$ *hcnj* (*w* ∗ *z*) = *hcnj w* ∗ *hcnj z*
⟨*proof*⟩

lemma *hcomplex-hcnj-divide*: $\bigwedge w\ z.$ *hcnj* (*w* / *z*) = *hcnj w* / *hcnj z*
⟨*proof*⟩

lemma *hcnj-one* [*simp*]: *hcnj 1* = *1*
⟨*proof*⟩

lemma *hcomplex-hcnj-zero* [*simp*]: *hcnj 0* = *0*
⟨*proof*⟩

lemma *hcomplex-hcnj-zero-iff* [*iff*]: $\bigwedge z.$ *hcnj z* = *0* ⟷ *z* = *0*
⟨*proof*⟩

lemma *hcomplex-mult-hcnj*: $\bigwedge z.$ *z* ∗ *hcnj z* = *hcomplex-of-hypreal* ((*hRe z*)^2 + (*hIm z*)^2)
⟨*proof*⟩

7.10 More Theorems about the Function *hcmod*

lemma *hcmod-hcomplex-of-hypreal-of-nat* [*simp*]:
hcmod (*hcomplex-of-hypreal* (*hypreal-of-nat n*)) = *hypreal-of-nat n*
⟨*proof*⟩

lemma *hcmod-hcomplex-of-hypreal-of-hypnat* [*simp*]:
hcmod (*hcomplex-of-hypreal*(*hypreal-of-hypnat n*)) = *hypreal-of-hypnat n*
⟨*proof*⟩

lemma *hcmod-mult-hcnj*: $\bigwedge z.$ *hcmod* (*z* ∗ *hcnj z*) = (*hcmod z*)^2
⟨*proof*⟩

lemma *hcmod-triangle-ineq2* [*simp*]: $\bigwedge a\ b.\ hcmod\ (b + a) - hcmod\ b \leq hcmod\ a$
⟨*proof*⟩

lemma *hcmod-diff-ineq* [*simp*]: $\bigwedge a\ b.\ hcmod\ a - hcmod\ b \leq hcmod\ (a + b)$
⟨*proof*⟩

7.11 Exponentiation

lemma *hcomplexpow-0* [*simp*]: $z\ \hat{}\ 0 = 1$
for z :: *hcomplex*
⟨*proof*⟩

lemma *hcomplexpow-Suc* [*simp*]: $z\ \hat{}\ (Suc\ n) = z * (z\ \hat{}\ n)$
for z :: *hcomplex*
⟨*proof*⟩

lemma *hcomplexpow-i-squared* [*simp*]: $iii^2 = -1$
⟨*proof*⟩

lemma *hcomplex-of-hypreal-pow*: $\bigwedge x.$ *hcomplex-of-hypreal* $(x\ \hat{}\ n) =$ *hcomplex-of-hypreal* $x\ \hat{}\ n$
⟨*proof*⟩

lemma *hcomplex-hcnj-pow*: $\bigwedge z.\ hcnj\ (z\ \hat{}\ n) = hcnj\ z\ \hat{}\ n$
⟨*proof*⟩

lemma *hcmod-hcomplexpow*: $\bigwedge x.\ hcmod\ (x\ \hat{}\ n) = hcmod\ x\ \hat{}\ n$
⟨*proof*⟩

lemma *hcpow-minus*:
$\bigwedge x\ n.\ (- x :: hcomplex)\ pow\ n = (if\ (\ *p*\ even)\ n\ then\ (x\ pow\ n)\ else - (x\ pow\ n))$
⟨*proof*⟩

lemma *hcpow-mult*: $(r * s)\ pow\ n = (r\ pow\ n) * (s\ pow\ n)$
for $r\ s$ :: *hcomplex*
⟨*proof*⟩

lemma *hcpow-zero2* [*simp*]: $\bigwedge n.\ 0\ pow\ (hSuc\ n) = (0::'a::semiring\text{-}1\ star)$
⟨*proof*⟩

lemma *hcpow-not-zero* [*simp,intro*]: $\bigwedge r\ n.\ r \neq 0 \Longrightarrow r\ pow\ n \neq (0::hcomplex)$
⟨*proof*⟩

lemma *hcpow-zero-zero*: $r\ pow\ n = 0 \Longrightarrow r = 0$
for r :: *hcomplex*
⟨*proof*⟩

7.12 The Function *hsgn*

lemma *hsgn-zero* [*simp*]: *hsgn 0 = 0*
⟨*proof*⟩

lemma *hsgn-one* [*simp*]: *hsgn 1 = 1*
⟨*proof*⟩

lemma *hsgn-minus*: $\bigwedge z.\ hsgn\ (-\ z) = -\ hsgn\ z$
⟨*proof*⟩

lemma *hsgn-eq*: $\bigwedge z.\ hsgn\ z = z\ /\ hcomplex\text{-}of\text{-}hypreal\ (hcmod\ z)$
⟨*proof*⟩

lemma *hcmod-i*: $\bigwedge x\ y.\ hcmod\ (HComplex\ x\ y) = (\ {*}f{*}\ sqrt)\ (x^2 + y^2)$
⟨*proof*⟩

lemma *hcomplex-eq-cancel-iff1* [*simp*]:
$hcomplex\text{-}of\text{-}hypreal\ xa = HComplex\ x\ y \longleftrightarrow xa = x \wedge y = 0$
⟨*proof*⟩

lemma *hcomplex-eq-cancel-iff2* [*simp*]:
$HComplex\ x\ y = hcomplex\text{-}of\text{-}hypreal\ xa \longleftrightarrow x = xa \wedge y = 0$
⟨*proof*⟩

lemma *HComplex-eq-0* [*simp*]: $\bigwedge x\ y.\ HComplex\ x\ y = 0 \longleftrightarrow x = 0 \wedge y = 0$
⟨*proof*⟩

lemma *HComplex-eq-1* [*simp*]: $\bigwedge x\ y.\ HComplex\ x\ y = 1 \longleftrightarrow x = 1 \wedge y = 0$
⟨*proof*⟩

lemma *i-eq-HComplex-0-1*: *iii = HComplex 0 1*
⟨*proof*⟩

lemma *HComplex-eq-i* [*simp*]: $\bigwedge x\ y.\ HComplex\ x\ y = iii \longleftrightarrow x = 0 \wedge y = 1$
⟨*proof*⟩

lemma *hRe-hsgn* [*simp*]: $\bigwedge z.\ hRe\ (hsgn\ z) = hRe\ z\ /\ hcmod\ z$
⟨*proof*⟩

lemma *hIm-hsgn* [*simp*]: $\bigwedge z.\ hIm\ (hsgn\ z) = hIm\ z\ /\ hcmod\ z$
⟨*proof*⟩

lemma *HComplex-inverse*: $\bigwedge x\ y.\ inverse\ (HComplex\ x\ y) = HComplex\ (x\ /\ (x^2 + y^2))\ (-\ y\ /\ (x^2 + y^2))$
⟨*proof*⟩

lemma *hRe-mult-i-eq*[*simp*]: $\bigwedge y.\ hRe\ (iii * hcomplex\text{-}of\text{-}hypreal\ y) = 0$
⟨*proof*⟩

lemma *hIm-mult-i-eq* [*simp*]: $\bigwedge y.\ hIm\ (iii * hcomplex\text{-}of\text{-}hypreal\ y) = y$
⟨*proof*⟩

lemma *hcmod-mult-i* [*simp*]: $\bigwedge y.\ hcmod\ (iii * hcomplex\text{-}of\text{-}hypreal\ y) = |y|$
⟨*proof*⟩

lemma *hcmod-mult-i2* [*simp*]: $\bigwedge y.\ hcmod\ (hcomplex\text{-}of\text{-}hypreal\ y * iii) = |y|$
⟨*proof*⟩

7.12.1 *harg*

lemma *cos-harg-i-mult-zero* [*simp*]: $\bigwedge y.\ y \neq 0 \Longrightarrow (\ *f*\ cos)\ (harg\ (HComplex\ 0\ y)) = 0$
⟨*proof*⟩

7.13 Polar Form for Nonstandard Complex Numbers

lemma *complex-split-polar2*: $\forall n.\ \exists r\ a.\ (z\ n) = complex\text{-}of\text{-}real\ r * Complex\ (cos\ a)\ (sin\ a)$
⟨*proof*⟩

lemma *hcomplex-split-polar*:
$\bigwedge z.\ \exists r\ a.\ z = hcomplex\text{-}of\text{-}hypreal\ r * (HComplex\ ((\ *f*\ cos)\ a)\ ((\ *f*\ sin)\ a))$
⟨*proof*⟩

lemma *hcis-eq*:
$\bigwedge a.\ hcis\ a = hcomplex\text{-}of\text{-}hypreal\ ((\ *f*\ cos)\ a) + iii * hcomplex\text{-}of\text{-}hypreal\ ((\ *f*\ sin)\ a)$
⟨*proof*⟩

lemma *hrcis-Ex*: $\bigwedge z.\ \exists r\ a.\ z = hrcis\ r\ a$
⟨*proof*⟩

lemma *hRe-hcomplex-polar* [*simp*]:
$\bigwedge r\ a.\ hRe\ (hcomplex\text{-}of\text{-}hypreal\ r * HComplex\ ((\ *f*\ cos)\ a)\ ((\ *f*\ sin)\ a)) = r * (\ *f*\ cos)\ a$
⟨*proof*⟩

lemma *hRe-hrcis* [*simp*]: $\bigwedge r\ a.\ hRe\ (hrcis\ r\ a) = r * (\ *f*\ cos)\ a$
⟨*proof*⟩

lemma *hIm-hcomplex-polar* [*simp*]:
$\bigwedge r\ a.\ hIm\ (hcomplex\text{-}of\text{-}hypreal\ r * HComplex\ ((\ *f*\ cos)\ a)\ ((\ *f*\ sin)\ a)) = r * (\ *f*\ sin)\ a$
⟨*proof*⟩

lemma *hIm-hrcis* [*simp*]: $\bigwedge r\ a.\ hIm\ (hrcis\ r\ a) = r * (\ *f*\ sin)\ a$
⟨*proof*⟩

lemma *hcmod-unit-one* [*simp*]: $\bigwedge a.$ *hcmod* (*HComplex* ((∗*f*∗ *cos*) *a*) ((∗*f*∗ *sin*) *a*)) = *1*
⟨*proof*⟩

lemma *hcmod-complex-polar* [*simp*]:
$\bigwedge r\ a.$ *hcmod* (*hcomplex-of-hypreal r* ∗ *HComplex* ((∗*f*∗ *cos*) *a*) ((∗*f*∗ *sin*) *a*)) = |*r*|
⟨*proof*⟩

lemma *hcmod-hrcis* [*simp*]: $\bigwedge r\ a.$ *hcmod*(*hrcis r a*) = |*r*|
⟨*proof*⟩

(*r1* ∗ *hrcis a*) ∗ (*r2* ∗ *hrcis b*) = *r1* ∗ *r2* ∗ *hrcis* (*a* + *b*)

lemma *hcis-hrcis-eq*: $\bigwedge a.$ *hcis a* = *hrcis 1 a*
⟨*proof*⟩
declare *hcis-hrcis-eq* [*symmetric*, *simp*]

lemma *hrcis-mult*: $\bigwedge a\ b\ r1\ r2.$ *hrcis r1 a* ∗ *hrcis r2 b* = *hrcis* (*r1* ∗ *r2*) (*a* + *b*)
⟨*proof*⟩

lemma *hcis-mult*: $\bigwedge a\ b.$ *hcis a* ∗ *hcis b* = *hcis* (*a* + *b*)
⟨*proof*⟩

lemma *hcis-zero* [*simp*]: *hcis 0* = *1*
⟨*proof*⟩

lemma *hrcis-zero-mod* [*simp*]: $\bigwedge a.$ *hrcis 0 a* = *0*
⟨*proof*⟩

lemma *hrcis-zero-arg* [*simp*]: $\bigwedge r.$ *hrcis r 0* = *hcomplex-of-hypreal r*
⟨*proof*⟩

lemma *hcomplex-i-mult-minus* [*simp*]: $\bigwedge x.$ *iii* ∗ (*iii* ∗ *x*) = − *x*
⟨*proof*⟩

lemma *hcomplex-i-mult-minus2* [*simp*]: *iii* ∗ *iii* ∗ *x* = − *x*
⟨*proof*⟩

lemma *hcis-hypreal-of-nat-Suc-mult*:
$\bigwedge a.$ *hcis* (*hypreal-of-nat* (*Suc n*) ∗ *a*) = *hcis a* ∗ *hcis* (*hypreal-of-nat n* ∗ *a*)
⟨*proof*⟩

lemma *NSDeMoivre*: $\bigwedge a.$ (*hcis a*) ˆ *n* = *hcis* (*hypreal-of-nat n* ∗ *a*)
⟨*proof*⟩

lemma *hcis-hypreal-of-hypnat-Suc-mult*:
$\bigwedge a\ n.$ *hcis* (*hypreal-of-hypnat* (*n* + *1*) ∗ *a*) = *hcis a* ∗ *hcis* (*hypreal-of-hypnat n* ∗ *a*)
⟨*proof*⟩

lemma *NSDeMoivre-ext*: $\bigwedge a\ n.\ (hcis\ a)\ pow\ n = hcis\ (hypreal\text{-}of\text{-}hypnat\ n * a)$
⟨*proof*⟩

lemma *NSDeMoivre2*: $\bigwedge a\ r.\ (hrcis\ r\ a)\ \hat{}\ n = hrcis\ (r\ \hat{}\ n)\ (hypreal\text{-}of\text{-}nat\ n * a)$
⟨*proof*⟩

lemma *DeMoivre2-ext*: $\bigwedge a\ r\ n.\ (hrcis\ r\ a)\ pow\ n = hrcis\ (r\ pow\ n)\ (hypreal\text{-}of\text{-}hypnat\ n * a)$
⟨*proof*⟩

lemma *hcis-inverse* [*simp*]: $\bigwedge a.\ inverse\ (hcis\ a) = hcis\ (-\ a)$
⟨*proof*⟩

lemma *hrcis-inverse*: $\bigwedge a\ r.\ inverse\ (hrcis\ r\ a) = hrcis\ (inverse\ r)\ (-\ a)$
⟨*proof*⟩

lemma *hRe-hcis* [*simp*]: $\bigwedge a.\ hRe\ (hcis\ a) = (\ *f*\ cos)\ a$
⟨*proof*⟩

lemma *hIm-hcis* [*simp*]: $\bigwedge a.\ hIm\ (hcis\ a) = (\ *f*\ sin)\ a$
⟨*proof*⟩

lemma *cos-n-hRe-hcis-pow-n*: $(\ *f*\ cos)\ (hypreal\text{-}of\text{-}nat\ n * a) = hRe\ (hcis\ a\ \hat{}\ n)$
⟨*proof*⟩

lemma *sin-n-hIm-hcis-pow-n*: $(\ *f*\ sin)\ (hypreal\text{-}of\text{-}nat\ n * a) = hIm\ (hcis\ a\ \hat{}\ n)$
⟨*proof*⟩

lemma *cos-n-hRe-hcis-hcpow-n*: $(\ *f*\ cos)\ (hypreal\text{-}of\text{-}hypnat\ n * a) = hRe\ (hcis\ a\ pow\ n)$
⟨*proof*⟩

lemma *sin-n-hIm-hcis-hcpow-n*: $(\ *f*\ sin)\ (hypreal\text{-}of\text{-}hypnat\ n * a) = hIm\ (hcis\ a\ pow\ n)$
⟨*proof*⟩

lemma *hExp-add*: $\bigwedge a\ b.\ hExp\ (a + b) = hExp\ a * hExp\ b$
⟨*proof*⟩

7.14 *hcomplex-of-complex*: the Injection from type *complex* to to *hcomplex*

lemma *hcomplex-of-complex-i*: *iii* = *hcomplex-of-complex* i
⟨*proof*⟩

lemma *hRe-hcomplex-of-complex*: *hRe* (*hcomplex-of-complex z*) = *hypreal-of-real* (*Re z*)
⟨*proof*⟩

lemma *hIm-hcomplex-of-complex*: *hIm* (*hcomplex-of-complex z*) = *hypreal-of-real* (*Im z*)
⟨*proof*⟩

lemma *hcmod-hcomplex-of-complex*: *hcmod* (*hcomplex-of-complex x*) = *hypreal-of-real* (*cmod x*)
⟨*proof*⟩

7.15 Numerals and Arithmetic

lemma *hcomplex-of-hypreal-eq-hcomplex-of-complex*:
hcomplex-of-hypreal (*hypreal-of-real x*) = *hcomplex-of-complex* (*complex-of-real x*)
⟨*proof*⟩

lemma *hcomplex-hypreal-numeral*:
hcomplex-of-complex (*numeral w*) = *hcomplex-of-hypreal*(*numeral w*)
⟨*proof*⟩

lemma *hcomplex-hypreal-neg-numeral*:
hcomplex-of-complex (− *numeral w*) = *hcomplex-of-hypreal*(− *numeral w*)
⟨*proof*⟩

lemma *hcomplex-numeral-hcnj* [*simp*]: *hcnj* (*numeral v* :: *hcomplex*) = *numeral v*
⟨*proof*⟩

lemma *hcomplex-numeral-hcmod* [*simp*]: *hcmod* (*numeral v* :: *hcomplex*) = (*numeral v* :: *hypreal*)
⟨*proof*⟩

lemma *hcomplex-neg-numeral-hcmod* [*simp*]: *hcmod* (− *numeral v* :: *hcomplex*) = (*numeral v* :: *hypreal*)
⟨*proof*⟩

lemma *hcomplex-numeral-hRe* [*simp*]: *hRe* (*numeral v* :: *hcomplex*) = *numeral v*
⟨*proof*⟩

lemma *hcomplex-numeral-hIm* [*simp*]: *hIm* (*numeral v* :: *hcomplex*) = *0*
⟨*proof*⟩

end

8 Star-Transforms in Non-Standard Analysis

theory *Star*

imports *NSA*
begin

definition — internal sets
starset-n :: $(nat \Rightarrow 'a\ set) \Rightarrow 'a\ star\ set$ $(*sn*$ - $[80]\ 80)$
where $*sn*\ As = Iset\ (star\text{-}n\ As)$

definition *InternalSets* :: $'a\ star\ set\ set$
where $InternalSets = \{X.\ \exists As.\ X = *sn*\ As\}$

definition — nonstandard extension of function
is-starext :: $('a\ star \Rightarrow 'a\ star) \Rightarrow ('a \Rightarrow 'a) \Rightarrow bool$
where *is-starext* $F\ f \longleftrightarrow$
$(\forall x\ y.\ \exists X \in Rep\text{-}star\ x.\ \exists Y \in Rep\text{-}star\ y.\ y = F\ x \longleftrightarrow eventually\ (\lambda n.\ Y\ n = f(X\ n))\ \mathcal{U})$

definition — internal functions
starfun-n :: $(nat \Rightarrow 'a \Rightarrow 'b) \Rightarrow 'a\ star \Rightarrow 'b\ star$ $(*fn*$ - $[80]\ 80)$
where $*fn*\ F = Ifun\ (star\text{-}n\ F)$

definition *InternalFuns* :: $('a\ star => 'b\ star)\ set$
where $InternalFuns = \{X.\ \exists F.\ X = *fn*\ F\}$

8.1 Preamble - Pulling $\exists$ over $\forall$

This proof does not need AC and was suggested by the referee for the JCM Paper: let $f\ x$ be least y such that $Q\ x\ y$.

lemma *no-choice*: $\forall x.\ \exists y.\ Q\ x\ y \Longrightarrow \exists f :: 'a \Rightarrow nat.\ \forall x.\ Q\ x\ (f\ x)$
⟨*proof*⟩

8.2 Properties of the Star-transform Applied to Sets of Reals

lemma *STAR-star-of-image-subset*: $star\text{-}of\ `\ A \subseteq *s*\ A$
⟨*proof*⟩

lemma *STAR-hypreal-of-real-Int*: $*s*\ X \cap \mathbb{R} = hypreal\text{-}of\text{-}real\ `\ X$
⟨*proof*⟩

lemma *STAR-star-of-Int*: $*s*\ X \cap Standard = star\text{-}of\ `\ X$
⟨*proof*⟩

lemma *lemma-not-hyprealA*: $x \notin hypreal\text{-}of\text{-}real\ `\ A \Longrightarrow \forall y \in A.\ x \neq hypreal\text{-}of\text{-}real\ y$
⟨*proof*⟩

lemma *lemma-not-starA*: $x \notin star\text{-}of\ `\ A \Longrightarrow \forall y \in A.\ x \neq star\text{-}of\ y$
⟨*proof*⟩

lemma *lemma-Compl-eq*: $-\ \{n.\ X\ n = xa\} = \{n.\ X\ n \neq xa\}$

⟨*proof*⟩

lemma *STAR-real-seq-to-hypreal*: $\forall n.\ (X\ n) \notin M \Longrightarrow$ *star-n* $X \notin *s*\ M$
⟨*proof*⟩

lemma *STAR-singleton*: $*s*\ \{x\} = \{$*star-of* $x\}$
⟨*proof*⟩

lemma *STAR-not-mem*: $x \notin F \Longrightarrow$ *star-of* $x \notin *s*\ F$
⟨*proof*⟩

lemma *STAR-subset-closed*: $x \in *s*\ A \Longrightarrow A \subseteq B \Longrightarrow x \in *s*\ B$
⟨*proof*⟩

Nonstandard extension of a set (defined using a constant sequence) as a special case of an internal set.

lemma *starset-n-starset*: $\forall n.\ As\ n = A \Longrightarrow *sn*\ As = *s*\ A$
⟨*proof*⟩

8.3 Theorems about nonstandard extensions of functions

Nonstandard extension of a function (defined using a constant sequence) as a special case of an internal function.

lemma *starfun-n-starfun*: $\forall n.\ F\ n = f \Longrightarrow *fn*\ F = *f*\ f$
⟨*proof*⟩

Prove that *abs* for hypreal is a nonstandard extension of abs for real w/o use of congruence property (proved after this for general nonstandard extensions of real valued functions).

Proof now Uses the ultrafilter tactic!

lemma *hrabs-is-starext-rabs*: *is-starext abs abs*
⟨*proof*⟩

Nonstandard extension of functions.

lemma *starfun*: $(\ *f*\ f)$ (*star-n* X) = *star-n* $(\lambda n.\ f\ (X\ n))$
⟨*proof*⟩

lemma *starfun-if-eq*: $\bigwedge w.\ w \neq$ *star-of* $x \Longrightarrow (\ *f*\ (\lambda z.$ *if* $z = x$ *then* a *else* $g\ z))$ $w = (\ *f*\ g)\ w$
⟨*proof*⟩

Multiplication: $(\ *f)\ x\ (\ *g) = *(f\ x\ g)$

lemma *starfun-mult*: $\bigwedge x.\ (\ *f*\ f)\ x * (\ *f*\ g)\ x = (\ *f*\ (\lambda x.\ f\ x * g\ x))\ x$
⟨*proof*⟩
declare *starfun-mult* [*symmetric*, *simp*]

Addition: $(\ *f) + (\ *g) = *(f + g)$

lemma *starfun-add*: $\bigwedge x.\ (\ *f*\ f)\ x + (\ *f*\ g)\ x = (\ *f*\ (\lambda x.\ f\ x + g\ x))\ x$
⟨*proof*⟩
declare *starfun-add* [*symmetric*, *simp*]

Subtraction: $(\ *f) + -(\ *g) = *(f + -g)$

lemma *starfun-minus*: $\bigwedge x.\ -\ (\ *f*\ f)\ x = (\ *f*\ (\lambda x.\ -\ f\ x))\ x$
⟨*proof*⟩
declare *starfun-minus* [*symmetric*, *simp*]

lemma *starfun-add-minus*: $\bigwedge x.\ (\ *f*\ f)\ x + -(\ *f*\ g)\ x = (\ *f*\ (\lambda x.\ f\ x + -g\ x))\ x$
⟨*proof*⟩
declare *starfun-add-minus* [*symmetric*, *simp*]

lemma *starfun-diff*: $\bigwedge x.\ (\ *f*\ f)\ x\ -\ (\ *f*\ g)\ x = (\ *f*\ (\lambda x.\ f\ x - g\ x))\ x$
⟨*proof*⟩
declare *starfun-diff* [*symmetric*, *simp*]

Composition: $(\ *f) \circ (\ *g) = *(f \circ g)$

lemma *starfun-o2*: $(\lambda x.\ (\ *f*\ f)\ ((\ *f*\ g)\ x)) = *f*\ (\lambda x.\ f\ (g\ x))$
⟨*proof*⟩

lemma *starfun-o*: $(\ *f*\ f) \circ (\ *f*\ g) = (\ *f*\ (f \circ g))$
⟨*proof*⟩

NS extension of constant function.

lemma *starfun-const-fun* [*simp*]: $\bigwedge x.\ (\ *f*\ (\lambda x.\ k))\ x =$ *star-of* k
⟨*proof*⟩

The NS extension of the identity function.

lemma *starfun-Id* [*simp*]: $\bigwedge x.\ (\ *f*\ (\lambda x.\ x))\ x = x$
⟨*proof*⟩

This is trivial, given *starfun-Id*.

lemma *starfun-Idfun-approx*: $x \approx$ *star-of* $a \Longrightarrow (\ *f*\ (\lambda x.\ x))\ x \approx$ *star-of* a
⟨*proof*⟩

The Star-function is a (nonstandard) extension of the function.

lemma *is-starext-starfun*: *is-starext* $(\ *f*\ f)\ f$
⟨*proof*⟩

Any nonstandard extension is in fact the Star-function.

lemma *is-starfun-starext*: *is-starext* $F\ f \Longrightarrow F = *f*\ f$
⟨*proof*⟩

lemma *is-starext-starfun-iff*: *is-starext* $F\ f \longleftrightarrow F = *f*\ f$
⟨*proof*⟩

Extented function has same solution as its standard version for real arguments. i.e they are the same for all real arguments.

lemma *starfun-eq*: $(*f* \ f) \ (star\text{-}of \ a) = star\text{-}of \ (f \ a)$
⟨*proof*⟩

lemma *starfun-approx*: $(*f* \ f) \ (star\text{-}of \ a) \approx star\text{-}of \ (f \ a)$
⟨*proof*⟩

Useful for NS definition of derivatives.

lemma *starfun-lambda-cancel*: $\bigwedge x'. \ (*f* \ (\lambda h. \ f \ (x + h))) \ x' = (*f* \ f) \ (star\text{-}of \ x + x')$
⟨*proof*⟩

lemma *starfun-lambda-cancel2*: $(*f* \ (\lambda h. \ f \ (g \ (x + h)))) \ x' = (*f* \ (f \circ g)) \ (star\text{-}of \ x + x')$
⟨*proof*⟩

lemma *starfun-mult-HFinite-approx*:
$(*f* \ f) \ x \approx l \Longrightarrow (*f* \ g) \ x \approx m \Longrightarrow l \in HFinite \Longrightarrow m \in HFinite \Longrightarrow$
$(*f* \ (\lambda x. \ f \ x * g \ x)) \ x \approx l * m$
for $l \ m$ :: $'a$::*real-normed-algebra star*
⟨*proof*⟩

lemma *starfun-add-approx*: $(*f* \ f) \ x \approx l \Longrightarrow (*f* \ g) \ x \approx m \Longrightarrow (*f* \ (\%x. \ f \ x + g \ x)) \ x \approx l + m$
⟨*proof*⟩

Examples: *hrabs* is nonstandard extension of *rabs*, *inverse* is nonstandard extension of *inverse*.

Can be proved easily using theorem *starfun* and properties of ultrafilter as for inverse below we use the theorem we proved above instead.

lemma *starfun-rabs-hrabs*: $*f* \ abs = abs$
⟨*proof*⟩

lemma *starfun-inverse-inverse* [*simp*]: $(*f* \ inverse) \ x = inverse \ x$
⟨*proof*⟩

lemma *starfun-inverse*: $\bigwedge x. \ inverse \ ((*f* \ f) \ x) = (*f* \ (\lambda x. \ inverse \ (f \ x))) \ x$
⟨*proof*⟩
declare *starfun-inverse* [*symmetric*, *simp*]

lemma *starfun-divide*: $\bigwedge x. \ (*f* \ f) \ x \ / \ (*f* \ g) \ x = (*f* \ (\lambda x. \ f \ x \ / \ g \ x)) \ x$
⟨*proof*⟩
declare *starfun-divide* [*symmetric*, *simp*]

lemma *starfun-inverse2*: $\bigwedge x. \ inverse \ ((*f* \ f) \ x) = (*f* \ (\lambda x. \ inverse \ (f \ x))) \ x$
⟨*proof*⟩

General lemma/theorem needed for proofs in elementary topology of the reals.

lemma *starfun-mem-starset*: $\bigwedge x.\ (\ *f*\ f)\ x : *s*\ A \Longrightarrow x \in *s*\ \{x.\ f\ x \in A\}$
⟨*proof*⟩

Alternative definition for *hrabs* with *rabs* function applied entrywise to equivalence class representative. This is easily proved using *starfun* and ns extension thm.

lemma *hypreal-hrabs*: $|star\text{-}n\ X| = star\text{-}n\ (\lambda n.\ |X\ n|)$
⟨*proof*⟩

Nonstandard extension of set through nonstandard extension of *rabs* function i.e. *hrabs*. A more general result should be where we replace *rabs* by some arbitrary function *f* and *hrabs* by its NS extenson. See second NS set extension below.

lemma *STAR-rabs-add-minus*: $*s*\ \{x.\ |x + -\ y| < r\} = \{x.\ |x + -star\text{-}of\ y| < star\text{-}of\ r\}$
⟨*proof*⟩

lemma *STAR-starfun-rabs-add-minus*:
$*s*\ \{x.\ |f\ x + -\ y| < r\} = \{x.\ |(\ *f*\ f)\ x + -star\text{-}of\ y| < star\text{-}of\ r\}$
⟨*proof*⟩

Another characterization of Infinitesimal and one of ≈ relation. In this theory since *hypreal-hrabs* proved here. Maybe move both theorems??

lemma *Infinitesimal-FreeUltrafilterNat-iff2*:
$star\text{-}n\ X \in Infinitesimal \longleftrightarrow (\forall m.\ eventually\ (\lambda n.\ norm\ (X\ n) < inverse\ (real\ (Suc\ m)))\ \mathcal{U})$
⟨*proof*⟩

lemma *HNatInfinite-inverse-Infinitesimal* [*simp*]:
$n \in HNatInfinite \Longrightarrow inverse\ (hypreal\text{-}of\text{-}hypnat\ n) \in Infinitesimal$
⟨*proof*⟩

lemma *approx-FreeUltrafilterNat-iff*:
$star\text{-}n\ X \approx star\text{-}n\ Y \longleftrightarrow (\forall r>0.\ eventually\ (\lambda n.\ norm\ (X\ n - Y\ n) < r)\ \mathcal{U})$
⟨*proof*⟩

lemma *approx-FreeUltrafilterNat-iff2*:
$star\text{-}n\ X \approx star\text{-}n\ Y \longleftrightarrow (\forall m.\ eventually\ (\lambda n.\ norm\ (X\ n - Y\ n) < inverse\ (real\ (Suc\ m)))\ \mathcal{U})$
⟨*proof*⟩

lemma *inj-starfun*: *inj starfun*
⟨*proof*⟩

end

9 Star-transforms for the Hypernaturals

theory *NatStar*
imports *Star*
begin

lemma *star-n-eq-starfun-whn*: *star-n* X = ($*f*$ X) *whn*
⟨*proof*⟩

lemma *starset-n-Un*: $*sn*$ (λn. (A n) ∪ (B n)) = $*sn*$ A ∪ $*sn*$ B
⟨*proof*⟩

lemma *InternalSets-Un*: X ∈ *InternalSets* ⟹ Y ∈ *InternalSets* ⟹ X ∪ Y ∈ *InternalSets*
⟨*proof*⟩

lemma *starset-n-Int*: $*sn*$ (λn. A n ∩ B n) = $*sn*$ A ∩ $*sn*$ B
⟨*proof*⟩

lemma *InternalSets-Int*: X ∈ *InternalSets* ⟹ Y ∈ *InternalSets* ⟹ X ∩ Y ∈ *InternalSets*
⟨*proof*⟩

lemma *starset-n-Compl*: $*sn*$ ((λn. − A n)) = − ($*sn*$ A)
⟨*proof*⟩

lemma *InternalSets-Compl*: X ∈ *InternalSets* ⟹ − X ∈ *InternalSets*
⟨*proof*⟩

lemma *starset-n-diff*: $*sn*$ (λn. (A n) − (B n)) = $*sn*$ A − $*sn*$ B
⟨*proof*⟩

lemma *InternalSets-diff*: X ∈ *InternalSets* ⟹ Y ∈ *InternalSets* ⟹ X − Y ∈ *InternalSets*
⟨*proof*⟩

lemma *NatStar-SHNat-subset*: *Nats* ≤ $*s*$ (*UNIV*:: *nat set*)
⟨*proof*⟩

lemma *NatStar-hypreal-of-real-Int*: $*s*$ X *Int Nats* = *hypnat-of-nat* ‘ X
⟨*proof*⟩

lemma *starset-starset-n-eq*: $*s*$ X = $*sn*$ (λn. X)
⟨*proof*⟩

lemma *InternalSets-starset-n* [*simp*]: ($*s*$ X) ∈ *InternalSets*
⟨*proof*⟩

lemma *InternalSets-UNIV-diff*: X ∈ *InternalSets* ⟹ *UNIV* − X ∈ *InternalSets*

⟨*proof*⟩

9.1 Nonstandard Extensions of Functions

Example of transfer of a property from reals to hyperreals — used for limit comparison of sequences.

lemma *starfun-le-mono*: $\forall n.\ N \leq n \longrightarrow f\ n \leq g\ n \Longrightarrow$
$\forall n.\ hypnat\text{-}of\text{-}nat\ N \leq n \longrightarrow (\ {*}f{*}\ f)\ n \leq (\ {*}f{*}\ g)\ n$
⟨*proof*⟩

And another:

lemma *starfun-less-mono*:
$\forall n.\ N \leq n \longrightarrow f\ n < g\ n \Longrightarrow \forall n.\ hypnat\text{-}of\text{-}nat\ N \leq n \longrightarrow (\ {*}f{*}\ f)\ n < (\ {*}f{*}\ g)\ n$
⟨*proof*⟩

Nonstandard extension when we increment the argument by one.

lemma *starfun-shift-one*: $\bigwedge N.\ (\ {*}f{*}\ (\lambda n.\ f\ (Suc\ n)))\ N = (\ {*}f{*}\ f)\ (N + (1{::}hypnat))$
⟨*proof*⟩

Nonstandard extension with absolute value.

lemma *starfun-abs*: $\bigwedge N.\ (\ {*}f{*}\ (\lambda n.\ |f\ n|))\ N = |(\ {*}f{*}\ f)\ N|$
⟨*proof*⟩

The *hyperpow* function as a nonstandard extension of *realpow*.

lemma *starfun-pow*: $\bigwedge N.\ (\ {*}f{*}\ (\lambda n.\ r\ \hat{}\ n))\ N = hypreal\text{-}of\text{-}real\ r\ pow\ N$
⟨*proof*⟩

lemma *starfun-pow2*: $\bigwedge N.\ (\ {*}f{*}\ (\lambda n.\ X\ n\ \hat{}\ m))\ N = (\ {*}f{*}\ X)\ N\ pow\ hypnat\text{-}of\text{-}nat\ m$
⟨*proof*⟩

lemma *starfun-pow3*: $\bigwedge R.\ (\ {*}f{*}\ (\lambda r.\ r\ \hat{}\ n))\ R = R\ pow\ hypnat\text{-}of\text{-}nat\ n$
⟨*proof*⟩

The *hypreal-of-hypnat* function as a nonstandard extension of *real*.

lemma *starfunNat-real-of-nat*: $(\ {*}f{*}\ real) = hypreal\text{-}of\text{-}hypnat$
⟨*proof*⟩

lemma *starfun-inverse-real-of-nat-eq*:
$N \in HNatInfinite \Longrightarrow (\ {*}f{*}\ (\lambda x{::}nat.\ inverse\ (real\ x)))\ N = inverse\ (hypreal\text{-}of\text{-}hypnat\ N)$
⟨*proof*⟩

Internal functions – some redundancy with ${*}f{*}$ now.

lemma *starfun-n*: $(\ {*}fn{*}\ f)\ (star\text{-}n\ X) = star\text{-}n\ (\lambda n.\ f\ n\ (X\ n))$
⟨*proof*⟩

Multiplication: $(\ {*}fn)\ x\ (\ {*}gn) = {*}(fn\ x\ gn)$

lemma *starfun-n-mult*: $(\ {*}fn{*}\ f)\ z * (\ {*}fn{*}\ g)\ z = (\ {*}fn{*}\ (\lambda i\ x.\ f\ i\ x * g\ i\ x))\ z$
⟨*proof*⟩

Addition: $(\ {*}fn) + (\ {*}gn) = {*}(fn + gn)$

lemma *starfun-n-add*: $(\ {*}fn{*}\ f)\ z + (\ {*}fn{*}\ g)\ z = (\ {*}fn{*}\ (\lambda i\ x.\ f\ i\ x + g\ i\ x))\ z$
⟨*proof*⟩

Subtraction: $(\ {*}fn) - (\ {*}gn) = {*}(fn + -\ gn)$

lemma *starfun-n-add-minus*: $(\ {*}fn{*}\ f)\ z + -(\ {*}fn{*}\ g)\ z = (\ {*}fn{*}\ (\lambda i\ x.\ f\ i\ x + -g\ i\ x))\ z$
⟨*proof*⟩

Composition: $(\ {*}fn) \circ (\ {*}gn) = {*}(fn \circ gn)$

lemma *starfun-n-const-fun* [*simp*]: $(\ {*}fn{*}\ (\lambda i\ x.\ k))\ z = star\text{-}of\ k$
⟨*proof*⟩

lemma *starfun-n-minus*: $-\ (\ {*}fn{*}\ f)\ x = (\ {*}fn{*}\ (\lambda i\ x.\ -\ (f\ i)\ x))\ x$
⟨*proof*⟩

lemma *starfun-n-eq* [*simp*]: $(\ {*}fn{*}\ f)\ (star\text{-}of\ n) = star\text{-}n\ (\lambda i.\ f\ i\ n)$
⟨*proof*⟩

lemma *starfun-eq-iff*: $((\ {*}f{*}\ f) = (\ {*}f{*}\ g)) \longleftrightarrow f = g$
⟨*proof*⟩

lemma *starfunNat-inverse-real-of-nat-Infinitesimal* [*simp*]:
$N \in HNatInfinite \Longrightarrow (\ {*}f{*}\ (\%x.\ inverse\ (real\ x)))\ N \in Infinitesimal$
⟨*proof*⟩

9.2 Nonstandard Characterization of Induction

lemma *hypnat-induct-obj*:
$\bigwedge n.\ ((\ {*}p{*}\ P)\ (0{::}hypnat) \wedge (\forall n.\ (\ {*}p{*}\ P)\ n \longrightarrow (\ {*}p{*}\ P)\ (n + 1))) \longrightarrow (\ {*}p{*}\ P)\ n$
⟨*proof*⟩

lemma *hypnat-induct*:
$\bigwedge n.\ (\ {*}p{*}\ P)\ (0{::}hypnat) \Longrightarrow (\bigwedge n.\ (\ {*}p{*}\ P)\ n \Longrightarrow (\ {*}p{*}\ P)\ (n + 1)) \Longrightarrow (\ {*}p{*}\ P)\ n$
⟨*proof*⟩

lemma *starP2-eq-iff*: $(\ {*}p2{*}\ (op\ =)) = (op\ =)$
⟨*proof*⟩

lemma *starP2-eq-iff2*: $(\ {*}p2{*}\ (\lambda x\ y.\ x = y))\ X\ Y \longleftrightarrow X = Y$
⟨*proof*⟩

lemma *nonempty-nat-set-Least-mem*: $c \in S \Longrightarrow (LEAST\ n.\ n \in S) \in S$
for S :: *nat set*
⟨*proof*⟩

lemma *nonempty-set-star-has-least*:
$\bigwedge S$::*nat set star*. *Iset* $S \neq \{\} \Longrightarrow \exists\, n \in$ *Iset* $S.\ \forall\, m \in$ *Iset* $S.\ n \leq m$
⟨*proof*⟩

lemma *nonempty-InternalNatSet-has-least*: $S \in$ *InternalSets* $\Longrightarrow S \neq \{\} \Longrightarrow \exists\, n \in S.\ \forall\, m \in S.\ n \leq m$
for S :: *hypnat set*
⟨*proof*⟩

Goldblatt, page 129 Thm 11.3.2.

lemma *internal-induct-lemma*:
$\bigwedge X$::*nat set star*.
(0::*hypnat*) $\in$ *Iset* $X \Longrightarrow \forall\, n.\ n \in$ *Iset* $X \longrightarrow n + 1 \in$ *Iset* $X \Longrightarrow$ *Iset* X = (*UNIV*:: *hypnat set*)
⟨*proof*⟩

lemma *internal-induct*:
$X \in$ *InternalSets* $\Longrightarrow$ (0::*hypnat*) $\in X \Longrightarrow \forall\, n.\ n \in X \longrightarrow n + 1 \in X \Longrightarrow X$ = (*UNIV*:: *hypnat set*)
⟨*proof*⟩

end

10 Sequences and Convergence (Nonstandard)

theory *HSEQ*
imports *HOL.Limits NatStar*
abbrevs ---> = $\longrightarrow_{NS}$
begin

definition *NSLIMSEQ* :: (*nat* ⇒ ′*a*::*real-normed-vector*) ⇒ ′*a* ⇒ *bool*
(((-)/ $\longrightarrow_{NS}$ (-)) [*60*, *60*] *60*) **where**
— Nonstandard definition of convergence of sequence
$X \longrightarrow_{NS} L \longleftrightarrow (\forall\, N \in$ *HNatInfinite*. (*f* X) $N \approx$ *star-of* L)

definition *nslim* :: (*nat* ⇒ ′*a*::*real-normed-vector*) ⇒ ′*a*
where *nslim* X = (*THE* $L.\ X \longrightarrow_{NS} L$)
— Nonstandard definition of limit using choice operator

definition *NSconvergent* :: (*nat* ⇒ ′*a*::*real-normed-vector*) ⇒ *bool*
where *NSconvergent* $X \longleftrightarrow (\exists\, L.\ X \longrightarrow_{NS} L)$
— Nonstandard definition of convergence

definition *NSBseq* :: (*nat* ⇒ ′*a*::*real-normed-vector*) ⇒ *bool*

where *NSBseq* $X \longleftrightarrow (\forall N \in HNatInfinite.\ (\ *f*\ X)\ N \in HFinite)$
— Nonstandard definition for bounded sequence

definition *NSCauchy* :: $(nat \Rightarrow 'a::real\text{-}normed\text{-}vector) \Rightarrow bool$
where *NSCauchy* $X \longleftrightarrow (\forall M \in HNatInfinite.\ \forall N \in HNatInfinite.\ (\ *f*\ X)\ M \approx (\ *f*\ X)\ N)$
— Nonstandard definition

10.1 Limits of Sequences

lemma *NSLIMSEQ-iff*: $(X \longrightarrow_{NS} L) \longleftrightarrow (\forall N \in HNatInfinite.\ (\ *f*\ X)\ N \approx star\text{-}of\ L)$
⟨*proof*⟩

lemma *NSLIMSEQ-I*: $(\bigwedge N.\ N \in HNatInfinite \Longrightarrow starfun\ X\ N \approx star\text{-}of\ L) \Longrightarrow X \longrightarrow_{NS} L$
⟨*proof*⟩

lemma *NSLIMSEQ-D*: $X \longrightarrow_{NS} L \Longrightarrow N \in HNatInfinite \Longrightarrow starfun\ X\ N \approx star\text{-}of\ L$
⟨*proof*⟩

lemma *NSLIMSEQ-const*: $(\lambda n.\ k) \longrightarrow_{NS} k$
⟨*proof*⟩

lemma *NSLIMSEQ-add*: $X \longrightarrow_{NS} a \Longrightarrow Y \longrightarrow_{NS} b \Longrightarrow (\lambda n.\ X\ n + Y\ n) \longrightarrow_{NS} a + b$
⟨*proof*⟩

lemma *NSLIMSEQ-add-const*: $f \longrightarrow_{NS} a \Longrightarrow (\lambda n.\ f\ n + b) \longrightarrow_{NS} a + b$
⟨*proof*⟩

lemma *NSLIMSEQ-mult*: $X \longrightarrow_{NS} a \Longrightarrow Y \longrightarrow_{NS} b \Longrightarrow (\lambda n.\ X\ n * Y\ n) \longrightarrow_{NS} a * b$
for $a\ b :: 'a::real\text{-}normed\text{-}algebra$
⟨*proof*⟩

lemma *NSLIMSEQ-minus*: $X \longrightarrow_{NS} a \Longrightarrow (\lambda n.\ -\ X\ n) \longrightarrow_{NS} -\ a$
⟨*proof*⟩

lemma *NSLIMSEQ-minus-cancel*: $(\lambda n.\ -\ X\ n) \longrightarrow_{NS} -a \Longrightarrow X \longrightarrow_{NS} a$
⟨*proof*⟩

lemma *NSLIMSEQ-diff*: $X \longrightarrow_{NS} a \Longrightarrow Y \longrightarrow_{NS} b \Longrightarrow (\lambda n.\ X\ n - Y\ n) \longrightarrow_{NS} a - b$
⟨*proof*⟩

lemma *NSLIMSEQ-add-minus*: $X \longrightarrow_{NS} a \Longrightarrow Y \longrightarrow_{NS} b \Longrightarrow (\lambda n.\ X\ n + -\ Y\ n) \longrightarrow_{NS} a + -\ b$
⟨*proof*⟩

lemma *NSLIMSEQ-diff-const*: $f \longrightarrow_{NS} a \Longrightarrow (\lambda n.\ f\ n - b) \longrightarrow_{NS} a - b$
⟨*proof*⟩

lemma *NSLIMSEQ-inverse*: $X \longrightarrow_{NS} a \Longrightarrow a \neq 0 \Longrightarrow (\lambda n.\ inverse\ (X\ n)) \longrightarrow_{NS} inverse\ a$
for a :: *'a::real-normed-div-algebra*
⟨*proof*⟩

lemma *NSLIMSEQ-mult-inverse*: $X \longrightarrow_{NS} a \Longrightarrow Y \longrightarrow_{NS} b \Longrightarrow b \neq 0 \Longrightarrow (\lambda n.\ X\ n\ /\ Y\ n) \longrightarrow_{NS} a\ /\ b$
for $a\ b$ :: *'a::real-normed-field*
⟨*proof*⟩

lemma *starfun-hnorm*: $\bigwedge x.\ hnorm\ ((\ {*f*}\ f)\ x) = (\ {*f*}\ (\lambda x.\ norm\ (f\ x)))\ x$
⟨*proof*⟩

lemma *NSLIMSEQ-norm*: $X \longrightarrow_{NS} a \Longrightarrow (\lambda n.\ norm\ (X\ n)) \longrightarrow_{NS} norm\ a$
⟨*proof*⟩

Uniqueness of limit.

lemma *NSLIMSEQ-unique*: $X \longrightarrow_{NS} a \Longrightarrow X \longrightarrow_{NS} b \Longrightarrow a = b$
⟨*proof*⟩

lemma *NSLIMSEQ-pow* [*rule-format*]: $(X \longrightarrow_{NS} a) \longrightarrow ((\lambda n.\ (X\ n)\ \hat{}\ m) \longrightarrow_{NS} a\ \hat{}\ m)$
for a :: *'a::{real-normed-algebra,power}*
⟨*proof*⟩

We can now try and derive a few properties of sequences, starting with the limit comparison property for sequences.

lemma *NSLIMSEQ-le*: $f \longrightarrow_{NS} l \Longrightarrow g \longrightarrow_{NS} m \Longrightarrow \exists N.\ \forall n \geq N.\ f\ n \leq g\ n \Longrightarrow l \leq m$
for $l\ m$ :: *real*
⟨*proof*⟩

lemma *NSLIMSEQ-le-const*: $X \longrightarrow_{NS} r \Longrightarrow \forall n.\ a \leq X\ n \Longrightarrow a \leq r$
for $a\ r$ :: *real*
⟨*proof*⟩

lemma *NSLIMSEQ-le-const2*: $X \longrightarrow_{NS} r \Longrightarrow \forall n.\ X\ n \leq a \Longrightarrow r \leq a$
for $a\ r$ :: *real*
⟨*proof*⟩

Shift a convergent series by 1: By the equivalence between Cauchiness and convergence and because the successor of an infinite hypernatural is also infinite.

lemma *NSLIMSEQ-Suc*: $f \longrightarrow_{NS} l \Longrightarrow (\lambda n.\ f(Suc\ n)) \longrightarrow_{NS} l$
⟨*proof*⟩

lemma *NSLIMSEQ-imp-Suc*: $(\lambda n.\ f(Suc\ n)) \longrightarrow_{NS} l \Longrightarrow f \longrightarrow_{NS} l$
⟨*proof*⟩

lemma *NSLIMSEQ-Suc-iff*: $(\lambda n.\ f\ (Suc\ n)) \longrightarrow_{NS} l \longleftrightarrow f \longrightarrow_{NS} l$
⟨*proof*⟩

10.1.1 Equivalence of *LIMSEQ* and *NSLIMSEQ*

lemma *LIMSEQ-NSLIMSEQ*:
assumes X: $X \longrightarrow L$
shows $X \longrightarrow_{NS} L$
⟨*proof*⟩

lemma *NSLIMSEQ-LIMSEQ*:
assumes X: $X \longrightarrow_{NS} L$
shows $X \longrightarrow L$
⟨*proof*⟩

theorem *LIMSEQ-NSLIMSEQ-iff*: $f \longrightarrow L \longleftrightarrow f \longrightarrow_{NS} L$
⟨*proof*⟩

10.1.2 Derived theorems about *NSLIMSEQ*

We prove the NS version from the standard one, since the NS proof seems more complicated than the standard one above!

lemma *NSLIMSEQ-norm-zero*: $(\lambda n.\ norm\ (X\ n)) \longrightarrow_{NS} 0 \longleftrightarrow X \longrightarrow_{NS} 0$
⟨*proof*⟩

lemma *NSLIMSEQ-rabs-zero*: $(\lambda n.\ |f\ n|) \longrightarrow_{NS} 0 \longleftrightarrow f \longrightarrow_{NS} (0::real)$
⟨*proof*⟩

Generalization to other limits.

lemma *NSLIMSEQ-imp-rabs*: $f \longrightarrow_{NS} l \Longrightarrow (\lambda n.\ |f\ n|) \longrightarrow_{NS} |l|$
for $l :: real$
⟨*proof*⟩

lemma *NSLIMSEQ-inverse-zero*: $\forall\, y::real.\ \exists\, N.\ \forall\, n \geq N.\ y < f\ n \Longrightarrow (\lambda n.\ inverse\ (f\ n)) \longrightarrow_{NS} 0$
⟨*proof*⟩

lemma *NSLIMSEQ-inverse-real-of-nat*: $(\lambda n.\ inverse\ (real\ (Suc\ n))) \longrightarrow_{NS} 0$
⟨*proof*⟩

lemma *NSLIMSEQ-inverse-real-of-nat-add*: $(\lambda n.\ r + inverse\ (real\ (Suc\ n))) \longrightarrow_{NS} r$
⟨*proof*⟩

lemma *NSLIMSEQ-inverse-real-of-nat-add-minus*: $(\lambda n.\ r + -\ inverse\ (real\ (Suc\ n))) \longrightarrow_{NS} r$
⟨*proof*⟩

lemma *NSLIMSEQ-inverse-real-of-nat-add-minus-mult*:
$(\lambda n.\ r * (1 + -\ inverse\ (real\ (Suc\ n)))) \longrightarrow_{NS} r$
⟨*proof*⟩

10.2 Convergence

lemma *nslimI*: $X \longrightarrow_{NS} L \Longrightarrow nslim\ X = L$
⟨*proof*⟩

lemma *lim-nslim-iff*: $lim\ X = nslim\ X$
⟨*proof*⟩

lemma *NSconvergentD*: $NSconvergent\ X \Longrightarrow \exists L.\ X \longrightarrow_{NS} L$
⟨*proof*⟩

lemma *NSconvergentI*: $X \longrightarrow_{NS} L \Longrightarrow NSconvergent\ X$
⟨*proof*⟩

lemma *convergent-NSconvergent-iff*: $convergent\ X = NSconvergent\ X$
⟨*proof*⟩

lemma *NSconvergent-NSLIMSEQ-iff*: $NSconvergent\ X \longleftrightarrow X \longrightarrow_{NS} nslim\ X$
⟨*proof*⟩

10.3 Bounded Monotonic Sequences

lemma *NSBseqD*: $NSBseq\ X \Longrightarrow N \in HNatInfinite \Longrightarrow (\ {*}f{*}\ X)\ N \in HFinite$
⟨*proof*⟩

lemma *Standard-subset-HFinite*: $Standard \subseteq HFinite$
⟨*proof*⟩

lemma *NSBseqD2*: $NSBseq\ X \Longrightarrow (\ {*}f{*}\ X)\ N \in HFinite$
⟨*proof*⟩

lemma *NSBseqI*: $\forall N \in HNatInfinite.\ (\ {*}f{*}\ X)\ N \in HFinite \Longrightarrow NSBseq\ X$
⟨*proof*⟩

The standard definition implies the nonstandard definition.

lemma *Bseq-NSBseq*: $Bseq\ X \Longrightarrow NSBseq\ X$

⟨*proof*⟩

The nonstandard definition implies the standard definition.

lemma *SReal-less-omega*: $r \in \mathbb{R} \Longrightarrow r < \omega$
⟨*proof*⟩

lemma *NSBseq-Bseq*: *NSBseq X* $\Longrightarrow$ *Bseq X*
⟨*proof*⟩

Equivalence of nonstandard and standard definitions for a bounded sequence.

lemma *Bseq-NSBseq-iff*: *Bseq X* = *NSBseq X*
⟨*proof*⟩

A convergent sequence is bounded: Boundedness as a necessary condition for convergence. The nonstandard version has no existential, as usual.

lemma *NSconvergent-NSBseq*: *NSconvergent X* $\Longrightarrow$ *NSBseq X*
⟨*proof*⟩

Standard Version: easily now proved using equivalence of NS and standard definitions.

lemma *convergent-Bseq*: *convergent X* $\Longrightarrow$ *Bseq X*
for X :: *nat* $\Rightarrow$ *'b::real-normed-vector*
⟨*proof*⟩

10.3.1 Upper Bounds and Lubs of Bounded Sequences

lemma *NSBseq-isUb*: *NSBseq X* $\Longrightarrow \exists\, U$::*real*. *isUb UNIV* $\{x.\ \exists\, n.\ X\ n = x\}$ U
⟨*proof*⟩

lemma *NSBseq-isLub*: *NSBseq X* $\Longrightarrow \exists\, U$::*real*. *isLub UNIV* $\{x.\ \exists\, n.\ X\ n = x\}$ U
⟨*proof*⟩

10.3.2 A Bounded and Monotonic Sequence Converges

The best of both worlds: Easier to prove this result as a standard theorem and then use equivalence to "transfer" it into the equivalent nonstandard form if needed!

lemma *Bmonoseq-NSLIMSEQ*: $\forall\, n \geq m.\ X\ n = X\ m \Longrightarrow \exists\, L.\ X \longrightarrow_{NS} L$
⟨*proof*⟩

lemma *NSBseq-mono-NSconvergent*: *NSBseq X* $\Longrightarrow \forall\, m.\ \forall\, n \geq m.\ X\ m \leq X\ n \Longrightarrow$ *NSconvergent X*
for X :: *nat* $\Rightarrow$ *real*
⟨*proof*⟩

10.4 Cauchy Sequences

lemma *NSCauchyI*:
($\bigwedge M\ N.\ M \in HNatInfinite \Longrightarrow N \in HNatInfinite \Longrightarrow starfun\ X\ M \approx starfun\ X\ N) \Longrightarrow NSCauchy\ X$
⟨*proof*⟩

lemma *NSCauchyD*:
$NSCauchy\ X \Longrightarrow M \in HNatInfinite \Longrightarrow N \in HNatInfinite \Longrightarrow starfun\ X\ M \approx starfun\ X\ N$
⟨*proof*⟩

10.4.1 Equivalence Between NS and Standard

lemma *Cauchy-NSCauchy*:
assumes *X*: *Cauchy X*
shows *NSCauchy X*
⟨*proof*⟩

lemma *NSCauchy-Cauchy*:
assumes *X*: *NSCauchy X*
shows *Cauchy X*
⟨*proof*⟩

theorem *NSCauchy-Cauchy-iff*: $NSCauchy\ X = Cauchy\ X$
⟨*proof*⟩

10.4.2 Cauchy Sequences are Bounded

A Cauchy sequence is bounded – nonstandard version.

lemma *NSCauchy-NSBseq*: $NSCauchy\ X \Longrightarrow NSBseq\ X$
⟨*proof*⟩

10.4.3 Cauchy Sequences are Convergent

Equivalence of Cauchy criterion and convergence: We will prove this using our NS formulation which provides a much easier proof than using the standard definition. We do not need to use properties of subsequences such as boundedness, monotonicity etc... Compare with Harrison's corresponding proof in HOL which is much longer and more complicated. Of course, we do not have problems which he encountered with guessing the right instantiations for his 'espsilon-delta' proof(s) in this case since the NS formulations do not involve existential quantifiers.

lemma *NSconvergent-NSCauchy*: $NSconvergent\ X \Longrightarrow NSCauchy\ X$
⟨*proof*⟩

lemma *real-NSCauchy-NSconvergent*: $NSCauchy\ X \Longrightarrow NSconvergent\ X$
for $X :: nat \Rightarrow real$

⟨*proof*⟩

lemma *NSCauchy-NSconvergent*: *NSCauchy X* $\Longrightarrow$ *NSconvergent X*
for *X* :: *nat* $\Rightarrow$ *'a::banach*
⟨*proof*⟩

lemma *NSCauchy-NSconvergent-iff*: *NSCauchy X* = *NSconvergent X*
for *X* :: *nat* $\Rightarrow$ *'a::banach*
⟨*proof*⟩

10.5 Power Sequences

The sequence x^n tends to 0 if $(0::'a) \leq x$ and $x < (1::'a)$. Proof will use (NS) Cauchy equivalence for convergence and also fact that bounded and monotonic sequence converges.

We now use NS criterion to bring proof of theorem through.

lemma *NSLIMSEQ-realpow-zero*: $0 \leq x \Longrightarrow x < 1 \Longrightarrow (\lambda n.\ x \hat{}\ n) \longrightarrow_{NS} 0$
for x :: *real*
⟨*proof*⟩

lemma *NSLIMSEQ-rabs-realpow-zero*: $|c| < 1 \Longrightarrow (\lambda n.\ |c| \hat{}\ n) \longrightarrow_{NS} 0$
for c :: *real*
⟨*proof*⟩

lemma *NSLIMSEQ-rabs-realpow-zero2*: $|c| < 1 \Longrightarrow (\lambda n.\ c \hat{}\ n) \longrightarrow_{NS} 0$
for c :: *real*
⟨*proof*⟩

end

11 Finite Summation and Infinite Series for Hyperreals

theory *HSeries*
imports *HSEQ*
begin

definition *sumhr* :: *hypnat* × *hypnat* × (*nat* $\Rightarrow$ *real*) $\Rightarrow$ *hypreal*
where *sumhr* = (λ(*M*,*N*,*f*). *starfun2* (λ*m n*. *sum f* {*m*..<*n*}) *M N*)

definition *NSsums* :: (*nat* $\Rightarrow$ *real*) $\Rightarrow$ *real* $\Rightarrow$ *bool* (**infixr** *NSsums 80*)
where *f NSsums s* = (λ*n*. *sum f* {..<*n*}) $\longrightarrow_{NS}$ *s*

definition *NSsummable* :: (*nat* $\Rightarrow$ *real*) $\Rightarrow$ *bool*
where *NSsummable f* $\longleftrightarrow$ ($\exists$ *s*. *f NSsums s*)

definition *NSsuminf* :: (*nat* ⇒ *real*) ⇒ *real*
where *NSsuminf f* = (*THE s. f NSsums s*)

lemma *sumhr-app*: *sumhr* (*M*, *N*, *f*) = (**f2** (λ*m n. sum f* {*m*..<*n*})) *M N*
⟨*proof*⟩

Base case in definition of *sumr*.

lemma *sumhr-zero* [*simp*]: ⋀*m. sumhr* (*m*, *0*, *f*) = *0*
⟨*proof*⟩

Recursive case in definition of *sumr*.

lemma *sumhr-if*:
⋀*m n. sumhr* (*m*, *n* + *1*, *f*) = (*if n* + *1* ≤ *m then 0 else sumhr* (*m*, *n*, *f*) + (**f** *f*) *n*)
⟨*proof*⟩

lemma *sumhr-Suc-zero* [*simp*]: ⋀*n. sumhr* (*n* + *1*, *n*, *f*) = *0*
⟨*proof*⟩

lemma *sumhr-eq-bounds* [*simp*]: ⋀*n. sumhr* (*n*, *n*, *f*) = *0*
⟨*proof*⟩

lemma *sumhr-Suc* [*simp*]: ⋀*m. sumhr* (*m*, *m* + *1*, *f*) = (**f** *f*) *m*
⟨*proof*⟩

lemma *sumhr-add-lbound-zero* [*simp*]: ⋀*k m. sumhr* (*m* + *k*, *k*, *f*) = *0*
⟨*proof*⟩

lemma *sumhr-add*: ⋀*m n. sumhr* (*m*, *n*, *f*) + *sumhr* (*m*, *n*, *g*) = *sumhr* (*m*, *n*, λ*i. f i* + *g i*)
⟨*proof*⟩

lemma *sumhr-mult*: ⋀*m n. hypreal-of-real r* ∗ *sumhr* (*m*, *n*, *f*) = *sumhr* (*m*, *n*, λ*n. r* ∗ *f n*)
⟨*proof*⟩

lemma *sumhr-split-add*: ⋀*n p. n* < *p* ⟹ *sumhr* (*0*, *n*, *f*) + *sumhr* (*n*, *p*, *f*) = *sumhr* (*0*, *p*, *f*)
⟨*proof*⟩

lemma *sumhr-split-diff*: *n* < *p* ⟹ *sumhr* (*0*, *p*, *f*) − *sumhr* (*0*, *n*, *f*) = *sumhr* (*n*, *p*, *f*)
⟨*proof*⟩

lemma *sumhr-hrabs*: ⋀*m n.* |*sumhr* (*m*, *n*, *f*)| ≤ *sumhr* (*m*, *n*, λ*i.* |*f i*|)
⟨*proof*⟩

Other general version also needed.

lemma *sumhr-fun-hypnat-eq*:
$(\forall r.\ m \leq r \wedge r < n \longrightarrow f\ r = g\ r) \longrightarrow$
$\quad sumhr\ (hypnat\text{-}of\text{-}nat\ m,\ hypnat\text{-}of\text{-}nat\ n,\ f) =$
$\quad sumhr\ (hypnat\text{-}of\text{-}nat\ m,\ hypnat\text{-}of\text{-}nat\ n,\ g)$
⟨*proof*⟩

lemma *sumhr-const*: $\bigwedge n.\ sumhr\ (0,\ n,\ \lambda i.\ r) = hypreal\text{-}of\text{-}hypnat\ n * hypreal\text{-}of\text{-}real\ r$
⟨*proof*⟩

lemma *sumhr-less-bounds-zero* [*simp*]: $\bigwedge m\ n.\ n < m \Longrightarrow sumhr\ (m,\ n,\ f) = 0$
⟨*proof*⟩

lemma *sumhr-minus*: $\bigwedge m\ n.\ sumhr\ (m,\ n,\ \lambda i.\ -\ f\ i) = -\ sumhr\ (m,\ n,\ f)$
⟨*proof*⟩

lemma *sumhr-shift-bounds*:
$\bigwedge m\ n.\ sumhr\ (m + hypnat\text{-}of\text{-}nat\ k,\ n + hypnat\text{-}of\text{-}nat\ k,\ f) =$
$\quad sumhr\ (m,\ n,\ \lambda i.\ f\ (i + k))$
⟨*proof*⟩

11.1 Nonstandard Sums

Infinite sums are obtained by summing to some infinite hypernatural (such as *whn*).

lemma *sumhr-hypreal-of-hypnat-omega*: $sumhr\ (0,\ whn,\ \lambda i.\ 1) = hypreal\text{-}of\text{-}hypnat\ whn$
⟨*proof*⟩

lemma *sumhr-hypreal-omega-minus-one*: $sumhr(0,\ whn,\ \lambda i.\ 1) = \omega - 1$
⟨*proof*⟩

lemma *sumhr-minus-one-realpow-zero* [*simp*]: $\bigwedge N.\ sumhr\ (0,\ N + N,\ \lambda i.\ (-1)\ \hat{}\ (i + 1)) = 0$
⟨*proof*⟩

lemma *sumhr-interval-const*:
$(\forall n.\ m \leq Suc\ n \longrightarrow f\ n = r) \wedge m \leq na \Longrightarrow$
$\quad sumhr\ (hypnat\text{-}of\text{-}nat\ m,\ hypnat\text{-}of\text{-}nat\ na,\ f) = hypreal\text{-}of\text{-}nat\ (na - m) * hypreal\text{-}of\text{-}real\ r$
⟨*proof*⟩

lemma *starfunNat-sumr*: $\bigwedge N.\ (\ *f*\ (\lambda n.\ sum\ f\ \{0..<n\}))\ N = sumhr\ (0,\ N,\ f)$
⟨*proof*⟩

lemma *sumhr-hrabs-approx* [*simp*]: $sumhr\ (0,\ M,\ f) \approx sumhr\ (0,\ N,\ f) \Longrightarrow |sumhr\ (M,\ N,\ f)| \approx 0$
⟨*proof*⟩

11.2 Infinite sums: Standard and NS theorems

lemma *sums-NSsums-iff*: $f\ sums\ l \longleftrightarrow f\ NSsums\ l$
⟨*proof*⟩

lemma *summable-NSsummable-iff*: $summable\ f \longleftrightarrow NSsummable\ f$
⟨*proof*⟩

lemma *suminf-NSsuminf-iff*: $suminf\ f = NSsuminf\ f$
⟨*proof*⟩

lemma *NSsums-NSsummable*: $f\ NSsums\ l \Longrightarrow NSsummable\ f$
⟨*proof*⟩

lemma *NSsummable-NSsums*: $NSsummable\ f \Longrightarrow f\ NSsums\ (NSsuminf\ f)$
⟨*proof*⟩

lemma *NSsums-unique*: $f\ NSsums\ s \Longrightarrow s = NSsuminf\ f$
⟨*proof*⟩

lemma *NSseries-zero*: $\forall m.\ n \le Suc\ m \longrightarrow f\ m = 0 \Longrightarrow f\ NSsums\ (sum\ f\ \{..<n\})$
⟨*proof*⟩

lemma *NSsummable-NSCauchy*:
$NSsummable\ f \longleftrightarrow (\forall M \in HNatInfinite.\ \forall N \in HNatInfinite.\ |sumhr\ (M,\ N,\ f)| \approx 0)$
⟨*proof*⟩

Terms of a convergent series tend to zero.

lemma *NSsummable-NSLIMSEQ-zero*: $NSsummable\ f \Longrightarrow f \longrightarrow_{NS} 0$
⟨*proof*⟩

Nonstandard comparison test.

lemma *NSsummable-comparison-test*: $\exists N.\ \forall n.\ N \le n \longrightarrow |f\ n| \le g\ n \Longrightarrow NSsummable\ g \Longrightarrow NSsummable\ f$
⟨*proof*⟩

lemma *NSsummable-rabs-comparison-test*:
$\exists N.\ \forall n.\ N \le n \longrightarrow |f\ n| \le g\ n \Longrightarrow NSsummable\ g \Longrightarrow NSsummable\ (\lambda k.\ |f\ k|)$
⟨*proof*⟩

end

12 Limits and Continuity (Nonstandard)

theory *HLim*
imports *Star*
abbrevs ---> = −□→$_{NS}$
begin

Nonstandard Definitions.

definition *NSLIM* :: $('a::real\text{-}normed\text{-}vector \Rightarrow 'b::real\text{-}normed\text{-}vector) \Rightarrow 'a \Rightarrow 'b \Rightarrow bool$
$(((\text{-})/ -(\text{-})/\rightarrow_{NS} (\text{-})) [60, 0, 60] 60)$
where $f -a\rightarrow_{NS} L \longleftrightarrow (\forall x.\ x \neq star\text{-}of\ a \wedge x \approx star\text{-}of\ a \longrightarrow (*f* f)\ x \approx star\text{-}of\ L)$

definition *isNSCont* :: $('a::real\text{-}normed\text{-}vector \Rightarrow 'b::real\text{-}normed\text{-}vector) \Rightarrow 'a \Rightarrow bool$
where — NS definition dispenses with limit notions
$isNSCont\ f\ a \longleftrightarrow (\forall y.\ y \approx star\text{-}of\ a \longrightarrow (*f* f)\ y \approx star\text{-}of\ (f\ a))$

definition *isNSUCont* :: $('a::real\text{-}normed\text{-}vector \Rightarrow 'b::real\text{-}normed\text{-}vector) \Rightarrow bool$
where $isNSUCont\ f \longleftrightarrow (\forall x\ y.\ x \approx y \longrightarrow (*f* f)\ x \approx (*f* f)\ y)$

12.1 Limits of Functions

lemma *NSLIM-I*: $(\bigwedge x.\ x \neq star\text{-}of\ a \Longrightarrow x \approx star\text{-}of\ a \Longrightarrow starfun\ f\ x \approx star\text{-}of\ L) \Longrightarrow f -a\rightarrow_{NS} L$
⟨*proof*⟩

lemma *NSLIM-D*: $f -a\rightarrow_{NS} L \Longrightarrow x \neq star\text{-}of\ a \Longrightarrow x \approx star\text{-}of\ a \Longrightarrow starfun\ f\ x \approx star\text{-}of\ L$
⟨*proof*⟩

Proving properties of limits using nonstandard definition. The properties hold for standard limits as well!

lemma *NSLIM-mult*: $f -x\rightarrow_{NS} l \Longrightarrow g -x\rightarrow_{NS} m \Longrightarrow (\lambda x.\ f\ x * g\ x) -x\rightarrow_{NS} (l * m)$
for $l\ m :: 'a::real\text{-}normed\text{-}algebra$
⟨*proof*⟩

lemma *starfun-scaleR* [*simp*]: $starfun\ (\lambda x.\ f\ x *_R g\ x) = (\lambda x.\ scaleHR\ (starfun\ f\ x)\ (starfun\ g\ x))$
⟨*proof*⟩

lemma *NSLIM-scaleR*: $f -x\rightarrow_{NS} l \Longrightarrow g -x\rightarrow_{NS} m \Longrightarrow (\lambda x.\ f\ x *_R g\ x) -x\rightarrow_{NS} (l *_R m)$
⟨*proof*⟩

lemma *NSLIM-add*: $f -x\rightarrow_{NS} l \Longrightarrow g -x\rightarrow_{NS} m \Longrightarrow (\lambda x.\ f\ x + g\ x) -x\rightarrow_{NS} (l + m)$
⟨*proof*⟩

lemma *NSLIM-const* [*simp*]: $(\lambda x.\ k) -x\rightarrow_{NS} k$
⟨*proof*⟩

lemma *NSLIM-minus*: $f -a\rightarrow_{NS} L \Longrightarrow (\lambda x.\ - f\ x) -a\rightarrow_{NS} -L$
⟨*proof*⟩

lemma *NSLIM-diff*: $f\ {-x}{\rightarrow}_{NS}\ l \Longrightarrow g\ {-x}{\rightarrow}_{NS}\ m \Longrightarrow (\lambda x.\ f\,x\ -\ g\ x)\ {-x}{\rightarrow}_{NS}\ (l\ -\ m)$
⟨*proof*⟩

lemma *NSLIM-add-minus*: $f\ {-x}{\rightarrow}_{NS}\ l \Longrightarrow g\ {-x}{\rightarrow}_{NS}\ m \Longrightarrow (\lambda x.\ f\,x\ +\ -\ g\ x)\ {-x}{\rightarrow}_{NS}\ (l\ +\ -m)$
⟨*proof*⟩

lemma *NSLIM-inverse*: $f\ {-a}{\rightarrow}_{NS}\ L \Longrightarrow L \neq 0 \Longrightarrow (\lambda x.\ inverse\ (f\,x))\ {-a}{\rightarrow}_{NS}\ (inverse\ L)$
for L :: *'a::real-normed-div-algebra*
⟨*proof*⟩

lemma *NSLIM-zero*:
assumes f: $f\ {-a}{\rightarrow}_{NS}\ l$
shows $(\lambda x.\ f(x)\ -\ l)\ {-a}{\rightarrow}_{NS}\ 0$
⟨*proof*⟩

lemma *NSLIM-zero-cancel*: $(\lambda x.\ f\,x\ -\ l)\ {-x}{\rightarrow}_{NS}\ 0 \Longrightarrow f\ {-x}{\rightarrow}_{NS}\ l$
⟨*proof*⟩

lemma *NSLIM-const-not-eq*: $k \neq L \Longrightarrow \neg\ (\lambda x.\ k)\ {-a}{\rightarrow}_{NS}\ L$
for a :: *'a::real-normed-algebra-1*
⟨*proof*⟩

lemma *NSLIM-not-zero*: $k \neq 0 \Longrightarrow \neg\ (\lambda x.\ k)\ {-a}{\rightarrow}_{NS}\ 0$
for a :: *'a::real-normed-algebra-1*
⟨*proof*⟩

lemma *NSLIM-const-eq*: $(\lambda x.\ k)\ {-a}{\rightarrow}_{NS}\ L \Longrightarrow k = L$
for a :: *'a::real-normed-algebra-1*
⟨*proof*⟩

lemma *NSLIM-unique*: $f\ {-a}{\rightarrow}_{NS}\ L \Longrightarrow f\ {-a}{\rightarrow}_{NS}\ M \Longrightarrow L = M$
for a :: *'a::real-normed-algebra-1*
⟨*proof*⟩

lemma *NSLIM-mult-zero*: $f\ {-x}{\rightarrow}_{NS}\ 0 \Longrightarrow g\ {-x}{\rightarrow}_{NS}\ 0 \Longrightarrow (\lambda x.\ f\ x\ *\ g\ x)\ {-x}{\rightarrow}_{NS}\ 0$
for $f\ g$ :: *'a::real-normed-vector* ⇒ *'b::real-normed-algebra*
⟨*proof*⟩

lemma *NSLIM-self*: $(\lambda x.\ x)\ {-a}{\rightarrow}_{NS}\ a$
⟨*proof*⟩

12.1.1 Equivalence of *filterlim* and *NSLIM*

lemma *LIM-NSLIM*:

assumes *f*: $f\ -a\rightarrow\ L$
shows $f\ -a\rightarrow_{NS}\ L$
⟨*proof*⟩

lemma *NSLIM-LIM*:
assumes *f*: $f\ -a\rightarrow_{NS}\ L$
shows $f\ -a\rightarrow\ L$
⟨*proof*⟩

theorem *LIM-NSLIM-iff*: $f\ -x\rightarrow\ L \longleftrightarrow f\ -x\rightarrow_{NS}\ L$
⟨*proof*⟩

12.2 Continuity

lemma *isNSContD*: *isNSCont f a* $\Longrightarrow y \approx$ *star-of a* $\Longrightarrow$ (**f* f*) $y \approx$ *star-of* (*f a*)
⟨*proof*⟩

lemma *isNSCont-NSLIM*: *isNSCont f a* $\Longrightarrow f\ -a\rightarrow_{NS}\ (f\ a)$
⟨*proof*⟩

lemma *NSLIM-isNSCont*: $f\ -a\rightarrow_{NS}\ (f\ a) \Longrightarrow$ *isNSCont f a*
⟨*proof*⟩

NS continuity can be defined using NS Limit in similar fashion to standard definition of continuity.

lemma *isNSCont-NSLIM-iff*: *isNSCont f a* $\longleftrightarrow f\ -a\rightarrow_{NS}\ (f\ a)$
⟨*proof*⟩

Hence, NS continuity can be given in terms of standard limit.

lemma *isNSCont-LIM-iff*: (*isNSCont f a*) = $(f\ -a\rightarrow\ (f\ a))$
⟨*proof*⟩

Moreover, it's trivial now that NS continuity is equivalent to standard continuity.

lemma *isNSCont-isCont-iff*: *isNSCont f a* $\longleftrightarrow$ *isCont f a*
⟨*proof*⟩

Standard continuity $\Longrightarrow$ NS continuity.

lemma *isCont-isNSCont*: *isCont f a* $\Longrightarrow$ *isNSCont f a*
⟨*proof*⟩

NS continuity $\Longrightarrow$ Standard continuity.

lemma *isNSCont-isCont*: *isNSCont f a* $\Longrightarrow$ *isCont f a*
⟨*proof*⟩

Alternative definition of continuity.

Prove equivalence between NS limits – seems easier than using standard definition.

lemma *NSLIM-h-iff*: $f\ -a\rightarrow_{NS}\ L \longleftrightarrow (\lambda h.\ f\ (a + h))\ -0\rightarrow_{NS}\ L$
⟨*proof*⟩

lemma *NSLIM-isCont-iff*: $f\ -a\rightarrow_{NS}\ f\ a \longleftrightarrow (\lambda h.\ f\ (a + h))\ -0\rightarrow_{NS}\ f\ a$
⟨*proof*⟩

lemma *isNSCont-minus*: *isNSCont f a* $\Longrightarrow$ *isNSCont* $(\lambda x.\ -\ f\ x)\ a$
⟨*proof*⟩

lemma *isNSCont-inverse*: *isNSCont f x* $\Longrightarrow f\ x \neq 0 \Longrightarrow$ *isNSCont* $(\lambda x.$ *inverse* $(f\ x))\ x$
for f :: *'a::real-normed-vector* ⇒ *'b::real-normed-div-algebra*
⟨*proof*⟩

lemma *isNSCont-const* [*simp*]: *isNSCont* $(\lambda x.\ k)\ a$
⟨*proof*⟩

lemma *isNSCont-abs* [*simp*]: *isNSCont abs a*
for a :: *real*
⟨*proof*⟩

12.3 Uniform Continuity

lemma *isNSUContD*: *isNSUCont f* $\Longrightarrow x \approx y \Longrightarrow (\ *f*\ f)\ x \approx (\ *f*\ f)\ y$
⟨*proof*⟩

lemma *isUCont-isNSUCont*:
fixes f :: *'a::real-normed-vector* ⇒ *'b::real-normed-vector*
assumes f: *isUCont f*
shows *isNSUCont f*
⟨*proof*⟩

lemma *isNSUCont-isUCont*:
fixes f :: *'a::real-normed-vector* ⇒ *'b::real-normed-vector*
assumes f: *isNSUCont f*
shows *isUCont f*
⟨*proof*⟩

end

13 Differentiation (Nonstandard)

theory *HDeriv*
imports *HLim*
begin

Nonstandard Definitions.

definition *nsderiv* :: ['a::real-normed-field ⇒ 'a, 'a, 'a] ⇒ bool
((NSDERIV (-)/ (-)/ :> (-)) [1000, 1000, 60] 60)
where *NSDERIV f x :> D* ⟷
(∀ h ∈ Infinitesimal − {0}. ((*f* f)(star-of x + h) − star-of (f x)) / h ≈ star-of D)

definition *NSdifferentiable* :: ['a::real-normed-field ⇒ 'a, 'a] ⇒ bool
(**infixl** *NSdifferentiable 60*)
where *f NSdifferentiable x* ⟷ (∃ D. NSDERIV f x :> D)

definition *increment* :: (real ⇒ real) ⇒ real ⇒ hypreal ⇒ hypreal
where *increment f x h* =
(SOME inc. f NSdifferentiable x ∧ inc = (*f* f) (hypreal-of-real x + h) − hypreal-of-real (f x))

13.1 Derivatives

lemma *DERIV-NS-iff*: (DERIV f x :> D) ⟷ (λh. (f (x + h) − f x) / h) $-0\rightarrow_{NS}$ D
⟨proof⟩

lemma *NS-DERIV-D*: DERIV f x :> D ⟹ (λh. (f (x + h) − f x) / h) $-0\rightarrow_{NS}$ D
⟨proof⟩

lemma *hnorm-of-hypreal*: ⋀r. hnorm ((*f* of-real) r::'a::real-normed-div-algebra star) = |r|
⟨proof⟩

lemma *Infinitesimal-of-hypreal*:
x ∈ Infinitesimal ⟹ ((*f* of-real) x::'a::real-normed-div-algebra star) ∈ Infinitesimal
⟨proof⟩

lemma *of-hypreal-eq-0-iff*: ⋀x. ((*f* of-real) x = (0::'a::real-algebra-1 star)) = (x = 0)
⟨proof⟩

lemma *NSDeriv-unique*: NSDERIV f x :> D ⟹ NSDERIV f x :> E ⟹ D = E
⟨proof⟩

First *NSDERIV* in terms of *NSLIM*.

First equivalence.

lemma *NSDERIV-NSLIM-iff*: (NSDERIV f x :> D) ⟷ (λh. (f (x + h) − f x) / h) $-0\rightarrow_{NS}$ D
⟨proof⟩

Second equivalence.

lemma *NSDERIV-NSLIM-iff2*: (*NSDERIV f x* :> *D*) ⟷ (λ*z*. (*f z* − *f x*) / (*z* − *x*)) −*x*→$_{NS}$ *D*
⟨*proof*⟩

While we're at it!

lemma *NSDERIV-iff2*:
(*NSDERIV f x* :> *D*) ⟷
(∀ *w*. *w* ≠ *star-of x* ∧ *w* ≈ *star-of x* ⟶ (*∗f∗* (λ*z*. (*f z* − *f x*) / (*z* − *x*))) *w* ≈ *star-of D*)
⟨*proof*⟩

lemma *hypreal-not-eq-minus-iff*: *x* ≠ *a* ⟷ *x* − *a* ≠ (*0*::′*a*::*ab-group-add*)
⟨*proof*⟩

lemma *NSDERIVD5*:
(*NSDERIV f x* :> *D*) ⟹
(∀ *u*. *u* ≈ *hypreal-of-real x* ⟶
(*∗f∗* (λ*z*. *f z* − *f x*)) *u* ≈ *hypreal-of-real D* ∗ (*u* − *hypreal-of-real x*))
⟨*proof*⟩

lemma *NSDERIVD4*:
(*NSDERIV f x* :> *D*) ⟹
(∀ *h* ∈ *Infinitesimal*.
(*∗f∗ f*)(*hypreal-of-real x* + *h*) − *hypreal-of-real* (*f x*) ≈ *hypreal-of-real D* ∗ *h*)
⟨*proof*⟩

lemma *NSDERIVD3*:
(*NSDERIV f x* :> *D*) ⟹
∀ *h* ∈ *Infinitesimal* − {*0*}.
((*∗f∗ f*) (*hypreal-of-real x* + *h*) − *hypreal-of-real* (*f x*)) ≈ *hypreal-of-real D* ∗ *h*
⟨*proof*⟩

Differentiability implies continuity nice and simple "algebraic" proof.

lemma *NSDERIV-isNSCont*: *NSDERIV f x* :> *D* ⟹ *isNSCont f x*
⟨*proof*⟩

Differentiation rules for combinations of functions follow from clear, straightforard, algebraic manipulations.

Constant function.

lemma *NSDERIV-const* [*simp*]: *NSDERIV* (λ*x*. *k*) *x* :> *0*
⟨*proof*⟩

Sum of functions- proved easily.

lemma *NSDERIV-add*:

NSDERIV f x :> Da $\Longrightarrow$ *NSDERIV g x :> Db* $\Longrightarrow$ *NSDERIV* ($\lambda x.$ *f x* + *g x*) *x :> Da* + *Db*
⟨*proof*⟩

Product of functions - Proof is trivial but tedious and long due to rearrangement of terms.

lemma *lemma-nsderiv1*: $(a * b) - (c * d) = (b * (a - c)) + (c * (b - d))$
for *a b c d* :: *'a::comm-ring star*
⟨*proof*⟩

lemma *lemma-nsderiv2*: $(x - y) / z =$ *star-of D* + *yb* $\Longrightarrow z \neq 0 \Longrightarrow$
$z \in$ *Infinitesimal* $\Longrightarrow$ *yb* $\in$ *Infinitesimal* $\Longrightarrow x - y \approx 0$
for *x y z* :: *'a::real-normed-field star*
⟨*proof*⟩

lemma *NSDERIV-mult*:
NSDERIV f x :> Da $\Longrightarrow$ *NSDERIV g x :> Db* $\Longrightarrow$
NSDERIV ($\lambda x.$ *f x* $*$ *g x*) *x :>* (*Da* $*$ *g x*) + (*Db* $*$ *f x*)
⟨*proof*⟩

Multiplying by a constant.

lemma *NSDERIV-cmult*: *NSDERIV f x :> D* $\Longrightarrow$ *NSDERIV* ($\lambda x.$ *c* $*$ *f x*) *x :>* *c* $*$ *D*
⟨*proof*⟩

Negation of function.

lemma *NSDERIV-minus*: *NSDERIV f x :> D* $\Longrightarrow$ *NSDERIV* ($\lambda x.$ $-$ *f x*) *x :>* $-$ *D*
⟨*proof*⟩

Subtraction.

lemma *NSDERIV-add-minus*:
NSDERIV f x :> Da $\Longrightarrow$ *NSDERIV g x :> Db* $\Longrightarrow$ *NSDERIV* ($\lambda x.$ *f x* + $-$ *g x*) *x :> Da* + $-$ *Db*
⟨*proof*⟩

lemma *NSDERIV-diff*:
NSDERIV f x :> Da $\Longrightarrow$ *NSDERIV g x :> Db* $\Longrightarrow$ *NSDERIV* ($\lambda x.$ *f x* $-$ *g x*) *x :> Da* $-$ *Db*
⟨*proof*⟩

Similarly to the above, the chain rule admits an entirely straightforward derivation. Compare this with Harrison's HOL proof of the chain rule, which proved to be trickier and required an alternative characterisation of differentiability- the so-called Carathedory derivative. Our main problem is manipulation of terms.

13.2 Lemmas

lemma *NSDERIV-zero*:
$NSDERIV\ g\ x :> D \Longrightarrow (\ {*f*}\ g)\ (star\text{-}of\ x + xa) = star\text{-}of\ (g\ x) \Longrightarrow$
$xa \in Infinitesimal \Longrightarrow xa \neq 0 \Longrightarrow D = 0$
⟨*proof*⟩

Can be proved differently using *NSLIM-isCont-iff*.

lemma *NSDERIV-approx*:
$NSDERIV\ f\ x :> D \Longrightarrow h \in Infinitesimal \Longrightarrow h \neq 0 \Longrightarrow$
$(\ {*f*}\ f)\ (star\text{-}of\ x + h) - star\text{-}of\ (f\ x) \approx 0$
⟨*proof*⟩

From one version of differentiability

$f\ x - f\ a$ −−−−−−−−−−−−−−− $\approx Db\ x - a$

lemma *NSDERIVD1*: $[|\ NSDERIV\ f\ (g\ x) :> Da;$
$(\ {*f*}\ g)\ (star\text{-}of(x) + xa) \neq star\text{-}of\ (g\ x);$
$(\ {*f*}\ g)\ (star\text{-}of(x) + xa) \approx star\text{-}of\ (g\ x)$
$|] ==> ((\ {*f*}\ f)\ ((\ {*f*}\ g)\ (star\text{-}of(x) + xa))$
$-\ star\text{-}of\ (f\ (g\ x)))$
$/\ ((\ {*f*}\ g)\ (star\text{-}of(x) + xa) - star\text{-}of\ (g\ x))$
$\approx star\text{-}of(Da)$
⟨*proof*⟩

From other version of differentiability

$f\ (x + h) - f\ x$ −−−−−−−−−−−−−−−−−−−− $\approx Db\ h$

lemma *NSDERIVD2*: $[|\ NSDERIV\ g\ x :> Db;\ xa \in Infinitesimal;\ xa \neq 0\ |]$
$==> ((\ {*f*}\ g)\ (star\text{-}of(x) + xa) - star\text{-}of(g\ x))\ /\ xa$
$\approx star\text{-}of(Db)$
⟨*proof*⟩

lemma *lemma-chain*: $z \neq 0 \Longrightarrow x * y = (x * inverse\ z) * (z * y)$
for $x\ y\ z$ :: *'a::real-normed-field star*
⟨*proof*⟩

This proof uses both definitions of differentiability.

lemma *NSDERIV-chain*:
$NSDERIV\ f\ (g\ x) :> Da \Longrightarrow NSDERIV\ g\ x :> Db \Longrightarrow NSDERIV\ (f \circ g)\ x :>$
$Da * Db$
⟨*proof*⟩

Differentiation of natural number powers.

lemma *NSDERIV-Id* [*simp*]: $NSDERIV\ (\lambda x.\ x)\ x :> 1$
⟨*proof*⟩

lemma *NSDERIV-cmult-Id* [*simp*]: $NSDERIV\ (op * c)\ x :> c$
⟨*proof*⟩

lemma *NSDERIV-inverse*:
fixes $x :: 'a::real\text{-}normed\text{-}field$
assumes $x \neq 0$ — can't get rid of $x \neq (0::'a)$ because it isn't continuous at zero

shows $NSDERIV\ (\lambda x.\ inverse\ x)\ x :> -\ (inverse\ x\ \hat{}\ Suc\ (Suc\ 0))$
⟨*proof*⟩

13.2.1 Equivalence of NS and Standard definitions

lemma *divideR-eq-divide*: $x /_R\ y = x\ /\ y$
⟨*proof*⟩

Now equivalence between *NSDERIV* and *DERIV*.

lemma *NSDERIV-DERIV-iff*: $NSDERIV\ f\ x :> D \longleftrightarrow DERIV\ f\ x :> D$
⟨*proof*⟩

NS version.

lemma *NSDERIV-pow*: $NSDERIV\ (\lambda x.\ x\ \hat{}\ n)\ x :> real\ n * (x\ \hat{}\ (n - Suc\ 0))$
⟨*proof*⟩

Derivative of inverse.

lemma *NSDERIV-inverse-fun*:
$NSDERIV\ f\ x :> d \Longrightarrow f\ x \neq 0 \Longrightarrow$
$NSDERIV\ (\lambda x.\ inverse\ (f\ x))\ x :> (-\ (d * inverse\ (f\ x\ \hat{}\ Suc\ (Suc\ 0))))$
for $x :: 'a::\{real\text{-}normed\text{-}field\}$
⟨*proof*⟩

Derivative of quotient.

lemma *NSDERIV-quotient*:
fixes $x :: 'a::real\text{-}normed\text{-}field$
shows $NSDERIV\ f\ x :> d \Longrightarrow NSDERIV\ g\ x :> e \Longrightarrow g\ x \neq 0 \Longrightarrow$
$NSDERIV\ (\lambda y.\ f\ y\ /\ g\ y)\ x :> (d * g\ x - (e * f\ x))\ /\ (g\ x\ \hat{}\ Suc\ (Suc\ 0))$
⟨*proof*⟩

lemma *CARAT-NSDERIV*:
$NSDERIV\ f\ x :> l \Longrightarrow \exists g.\ (\forall z.\ f\ z - f\ x = g\ z * (z - x)) \wedge isNSCont\ g\ x \wedge g$
$x = l$
⟨*proof*⟩

lemma *hypreal-eq-minus-iff3*: $x = y + z \longleftrightarrow x + -\ z = y$
for $x\ y\ z :: hypreal$
⟨*proof*⟩

lemma *CARAT-DERIVD*:
assumes *all*: $\forall z.\ f\ z - f\ x = g\ z * (z - x)$
and *nsc*: $isNSCont\ g\ x$
shows $NSDERIV\ f\ x :> g\ x$
⟨*proof*⟩

13.2.2 Differentiability predicate

lemma *NSdifferentiableD*: *f NSdifferentiable x* $\Longrightarrow \exists D.$ *NSDERIV f x* :> *D*
⟨*proof*⟩

lemma *NSdifferentiableI*: *NSDERIV f x* :> *D* $\Longrightarrow$ *f NSdifferentiable x*
⟨*proof*⟩

13.3 (NS) Increment

lemma *incrementI*:
f NSdifferentiable x $\Longrightarrow$
increment f x h = (*f* *f*) (*hypreal-of-real x* + *h*) − *hypreal-of-real* (*f x*)
⟨*proof*⟩

lemma *incrementI2*:
NSDERIV f x :> *D* $\Longrightarrow$
increment f x h = (*f* *f*) (*hypreal-of-real x* + *h*) − *hypreal-of-real* (*f x*)
⟨*proof*⟩

The Increment theorem – Keisler p. 65.

lemma *increment-thm*:
NSDERIV f x :> *D* $\Longrightarrow$ *h* ∈ *Infinitesimal* $\Longrightarrow$ *h* ≠ *0* $\Longrightarrow$
∃ *e* ∈ *Infinitesimal*. *increment f x h* = *hypreal-of-real D* ∗ *h* + *e* ∗ *h*
⟨*proof*⟩

lemma *increment-thm2*:
NSDERIV f x :> *D* $\Longrightarrow$ *h* ≈ *0* $\Longrightarrow$ *h* ≠ *0* $\Longrightarrow$
∃ *e* ∈ *Infinitesimal*. *increment f x h* = *hypreal-of-real D* ∗ *h* + *e* ∗ *h*
⟨*proof*⟩

lemma *increment-approx-zero*: *NSDERIV f x* :> *D* $\Longrightarrow$ *h* ≈ *0* $\Longrightarrow$ *h* ≠ *0* $\Longrightarrow$ *increment f x h* ≈ *0*
⟨*proof*⟩

end

14 Nonstandard Extensions of Transcendental Functions

theory *HTranscendental*
imports *HOL.Transcendental HSeries HDeriv*
begin

definition
exphr :: *real* => *hypreal* **where**
— define exponential function using standard part
exphr x = *st*(*sumhr* (*0*, *whn*, %*n*. *inverse* (*fact n*) ∗ (*x* ^ *n*)))

definition
sinhr :: *real* => *hypreal* **where**
sinhr x = *st*(*sumhr* (*0*, *whn*, %*n*. *sin-coeff n* ∗ *x* ^ *n*))

definition
coshr :: *real* => *hypreal* **where**
coshr x = *st*(*sumhr* (*0*, *whn*, %*n*. *cos-coeff n* ∗ *x* ^ *n*))

14.1 Nonstandard Extension of Square Root Function

lemma *STAR-sqrt-zero* [*simp*]: (∗*f*∗ *sqrt*) *0* = *0*
⟨*proof*⟩

lemma *STAR-sqrt-one* [*simp*]: (∗*f*∗ *sqrt*) *1* = *1*
⟨*proof*⟩

lemma *hypreal-sqrt-pow2-iff*: ((∗*f*∗ *sqrt*)(*x*) ^ *2* = *x*) = (*0* ≤ *x*)
⟨*proof*⟩

lemma *hypreal-sqrt-gt-zero-pow2*: !!*x*. *0* < *x* ==> (∗*f*∗ *sqrt*) (*x*) ^ *2* = *x*
⟨*proof*⟩

lemma *hypreal-sqrt-pow2-gt-zero*: *0* < *x* ==> *0* < (∗*f*∗ *sqrt*) (*x*) ^ *2*
⟨*proof*⟩

lemma *hypreal-sqrt-not-zero*: *0* < *x* ==> (∗*f*∗ *sqrt*) (*x*) ≠ *0*
⟨*proof*⟩

lemma *hypreal-inverse-sqrt-pow2*:
0 < *x* ==> *inverse* ((∗*f*∗ *sqrt*)(*x*)) ^ *2* = *inverse x*
⟨*proof*⟩

lemma *hypreal-sqrt-mult-distrib*:
!!*x y*. [|*0* < *x*; *0* <*y* |] ==>
(∗*f*∗ *sqrt*)(*x*∗*y*) = (∗*f*∗ *sqrt*)(*x*) ∗ (∗*f*∗ *sqrt*)(*y*)
⟨*proof*⟩

lemma *hypreal-sqrt-mult-distrib2*:
[|*0*≤*x*; *0*≤*y* |] ==>
(∗*f*∗ *sqrt*)(*x*∗*y*) = (∗*f*∗ *sqrt*)(*x*) ∗ (∗*f*∗ *sqrt*)(*y*)
⟨*proof*⟩

lemma *hypreal-sqrt-approx-zero* [*simp*]:
0 < *x* ==> ((∗*f*∗ *sqrt*)(*x*) ≈ *0*) = (*x* ≈ *0*)
⟨*proof*⟩

lemma *hypreal-sqrt-approx-zero2* [*simp*]:
0 ≤ *x* ==> ((∗*f*∗ *sqrt*)(*x*) ≈ *0*) = (*x* ≈ *0*)
⟨*proof*⟩

lemma *hypreal-sqrt-sum-squares* [*simp*]:
$((\ *f*\ sqrt)(x*x + y*y + z*z) \approx 0) = (x*x + y*y + z*z \approx 0)$
⟨*proof*⟩

lemma *hypreal-sqrt-sum-squares2* [*simp*]:
$((\ *f*\ sqrt)(x*x + y*y) \approx 0) = (x*x + y*y \approx 0)$
⟨*proof*⟩

lemma *hypreal-sqrt-gt-zero*: $!!x.\ 0 < x ==> 0 < (\ *f*\ sqrt)(x)$
⟨*proof*⟩

lemma *hypreal-sqrt-ge-zero*: $0 \le x ==> 0 \le (\ *f*\ sqrt)(x)$
⟨*proof*⟩

lemma *hypreal-sqrt-hrabs* [*simp*]: $!!x.\ (\ *f*\ sqrt)(x^2) = |x|$
⟨*proof*⟩

lemma *hypreal-sqrt-hrabs2* [*simp*]: $!!x.\ (\ *f*\ sqrt)(x*x) = |x|$
⟨*proof*⟩

lemma *hypreal-sqrt-hyperpow-hrabs* [*simp*]:
$!!x.\ (\ *f*\ sqrt)(x\ pow\ (hypnat\text{-}of\text{-}nat\ 2)) = |x|$
⟨*proof*⟩

lemma *star-sqrt-HFinite*: $[\![x \in HFinite;\ 0 \le x]\!] \Longrightarrow (\ *f*\ sqrt)\ x \in HFinite$
⟨*proof*⟩

lemma *st-hypreal-sqrt*:
$[|\ x \in HFinite;\ 0 \le x\ |] ==> st((\ *f*\ sqrt)\ x) = (\ *f*\ sqrt)(st\ x)$
⟨*proof*⟩

lemma *hypreal-sqrt-sum-squares-ge1* [*simp*]: $!!x\ y.\ x \le (\ *f*\ sqrt)(x^2 + y^2)$
⟨*proof*⟩

lemma *HFinite-hypreal-sqrt*:
$[|\ 0 \le x;\ x \in HFinite\ |] ==> (\ *f*\ sqrt)\ x \in HFinite$
⟨*proof*⟩

lemma *HFinite-hypreal-sqrt-imp-HFinite*:
$[|\ 0 \le x;\ (\ *f*\ sqrt)\ x \in HFinite\ |] ==> x \in HFinite$
⟨*proof*⟩

lemma *HFinite-hypreal-sqrt-iff* [*simp*]:
$0 \le x ==> ((\ *f*\ sqrt)\ x \in HFinite) = (x \in HFinite)$
⟨*proof*⟩

lemma *HFinite-sqrt-sum-squares* [*simp*]:
$((\ *f*\ sqrt)(x*x + y*y) \in HFinite) = (x*x + y*y \in HFinite)$

⟨proof⟩

lemma *Infinitesimal-hypreal-sqrt*:
[| 0 ≤ x; x ∈ Infinitesimal |] ==> (*f* sqrt) x ∈ Infinitesimal
⟨proof⟩

lemma *Infinitesimal-hypreal-sqrt-imp-Infinitesimal*:
[| 0 ≤ x; (*f* sqrt) x ∈ Infinitesimal |] ==> x ∈ Infinitesimal
⟨proof⟩

lemma *Infinitesimal-hypreal-sqrt-iff* [simp]:
0 ≤ x ==> ((*f* sqrt) x ∈ Infinitesimal) = (x ∈ Infinitesimal)
⟨proof⟩

lemma *Infinitesimal-sqrt-sum-squares* [simp]:
((*f* sqrt)(x*x + y*y) ∈ Infinitesimal) = (x*x + y*y ∈ Infinitesimal)
⟨proof⟩

lemma *HInfinite-hypreal-sqrt*:
[| 0 ≤ x; x ∈ HInfinite |] ==> (*f* sqrt) x ∈ HInfinite
⟨proof⟩

lemma *HInfinite-hypreal-sqrt-imp-HInfinite*:
[| 0 ≤ x; (*f* sqrt) x ∈ HInfinite |] ==> x ∈ HInfinite
⟨proof⟩

lemma *HInfinite-hypreal-sqrt-iff* [simp]:
0 ≤ x ==> ((*f* sqrt) x ∈ HInfinite) = (x ∈ HInfinite)
⟨proof⟩

lemma *HInfinite-sqrt-sum-squares* [simp]:
((*f* sqrt)(x*x + y*y) ∈ HInfinite) = (x*x + y*y ∈ HInfinite)
⟨proof⟩

lemma *HFinite-exp* [simp]:
sumhr (0, whn, %n. inverse (fact n) * x ^ n) ∈ HFinite
⟨proof⟩

lemma *exphr-zero* [simp]: exphr 0 = 1
⟨proof⟩

lemma *coshr-zero* [simp]: coshr 0 = 1
⟨proof⟩

lemma *STAR-exp-zero-approx-one* [simp]: (*f* exp) (0::hypreal) ≈ 1
⟨proof⟩

lemma *STAR-exp-Infinitesimal*: x ∈ Infinitesimal ==> (*f* exp) (x::hypreal) ≈ 1

⟨proof⟩

lemma *STAR-exp-epsilon* [*simp*]: (*f* exp) ε ≈ 1
⟨proof⟩

lemma *STAR-exp-add*:
!!(x::'a:: {banach,real-normed-field} star) y. (*f* exp)(x + y) = (*f* exp) x * (*f* exp) y
⟨proof⟩

lemma *exphr-hypreal-of-real-exp-eq*: exphr x = hypreal-of-real (exp x)
⟨proof⟩

lemma *starfun-exp-ge-add-one-self* [*simp*]: !!x::hypreal. 0 ≤ x ==> (1 + x) ≤ (*f* exp) x
⟨proof⟩

lemma *starfun-exp-HInfinite*:
[| x ∈ HInfinite; 0 ≤ x |] ==> (*f* exp) (x::hypreal) ∈ HInfinite
⟨proof⟩

lemma *starfun-exp-minus*:
!!x::'a:: {banach,real-normed-field} star. (*f* exp) (−x) = inverse((*f* exp) x)
⟨proof⟩

lemma *starfun-exp-Infinitesimal*:
[| x ∈ HInfinite; x ≤ 0 |] ==> (*f* exp) (x::hypreal) ∈ Infinitesimal
⟨proof⟩

lemma *starfun-exp-gt-one* [*simp*]: !!x::hypreal. 0 < x ==> 1 < (*f* exp) x
⟨proof⟩

abbreviation *real-ln* :: real ⇒ real **where**
real-ln ≡ ln

lemma *starfun-ln-exp* [*simp*]: !!x. (*f* real-ln) ((*f* exp) x) = x
⟨proof⟩

lemma *starfun-exp-ln-iff* [*simp*]: !!x. ((*f* exp)((*f* real-ln) x) = x) = (0 < x)
⟨proof⟩

lemma *starfun-exp-ln-eq*: !!u x. (*f* exp) u = x ==> (*f* real-ln) x = u
⟨proof⟩

lemma *starfun-ln-less-self* [*simp*]: !!x. 0 < x ==> (*f* real-ln) x < x
⟨proof⟩

lemma *starfun-ln-ge-zero* [*simp*]: !!*x*. *1* ≤ *x* ==> *0* ≤ (*f* *real-ln*) *x*
⟨*proof*⟩

lemma *starfun-ln-gt-zero* [*simp*]: !!*x* .*1* < *x* ==> *0* < (*f* *real-ln*) *x*
⟨*proof*⟩

lemma *starfun-ln-not-eq-zero* [*simp*]: !!*x*. [| *0* < *x*; *x* ≠ *1* |] ==> (*f* *real-ln*) *x* ≠ *0*
⟨*proof*⟩

lemma *starfun-ln-HFinite*: [| *x* ∈ *HFinite*; *1* ≤ *x* |] ==> (*f* *real-ln*) *x* ∈ *HFinite*
⟨*proof*⟩

lemma *starfun-ln-inverse*: !!*x*. *0* < *x* ==> (*f* *real-ln*) (*inverse x*) = −(*f* *ln*) *x*
⟨*proof*⟩

lemma *starfun-abs-exp-cancel*: ⋀*x*. |(*f* *exp*) (*x*::*hypreal*)| = (*f* *exp*) *x*
⟨*proof*⟩

lemma *starfun-exp-less-mono*: ⋀*x y*::*hypreal*. *x* < *y* ⟹ (*f* *exp*) *x* < (*f* *exp*) *y*
⟨*proof*⟩

lemma *starfun-exp-HFinite*: *x* ∈ *HFinite* ==> (*f* *exp*) (*x*::*hypreal*) ∈ *HFinite*
⟨*proof*⟩

lemma *starfun-exp-add-HFinite-Infinitesimal-approx*:
[|*x* ∈ *Infinitesimal*; *z* ∈ *HFinite* |] ==> (*f* *exp*) (*z* + *x*::*hypreal*) ≈ (*f* *exp*) *z*
⟨*proof*⟩

lemma *starfun-ln-HInfinite*:
[| *x* ∈ *HInfinite*; *0* < *x* |] ==> (*f* *real-ln*) *x* ∈ *HInfinite*
⟨*proof*⟩

lemma *starfun-exp-HInfinite-Infinitesimal-disj*:
x ∈ *HInfinite* ==> (*f* *exp*) *x* ∈ *HInfinite* | (*f* *exp*) (*x*::*hypreal*) ∈ *Infinitesimal*
⟨*proof*⟩

lemma *starfun-ln-HFinite-not-Infinitesimal*:
[| *x* ∈ *HFinite* − *Infinitesimal*; *0* < *x* |] ==> (*f* *real-ln*) *x* ∈ *HFinite*
⟨*proof*⟩

lemma *starfun-ln-Infinitesimal-HInfinite*:
[| *x* ∈ *Infinitesimal*; *0* < *x* |] ==> (*f* *real-ln*) *x* ∈ *HInfinite*

⟨proof⟩

lemma *starfun-ln-less-zero*: !!x. [| $0 < x$; $x < 1$ |] ==> (*f* real-ln) $x < 0$
⟨proof⟩

lemma *starfun-ln-Infinitesimal-less-zero*:
[| $x \in$ *Infinitesimal*; $0 < x$ |] ==> (*f* real-ln) $x < 0$
⟨proof⟩

lemma *starfun-ln-HInfinite-gt-zero*:
[| $x \in$ *HInfinite*; $0 < x$ |] ==> $0 <$ (*f* real-ln) x
⟨proof⟩

lemma *HFinite-sin* [*simp*]: *sumhr* (*0*, *whn*, %n. *sin-coeff* n * x ^ n) ∈ *HFinite*
⟨proof⟩

lemma *STAR-sin-zero* [*simp*]: (*f* sin) 0 = 0
⟨proof⟩

lemma *STAR-sin-Infinitesimal* [*simp*]:
fixes x :: 'a::{real-normed-field,banach} star
shows x ∈ *Infinitesimal* ==> (*f* sin) x ≈ x
⟨proof⟩

lemma *HFinite-cos* [*simp*]: *sumhr* (*0*, *whn*, %n. *cos-coeff* n * x ^ n) ∈ *HFinite*
⟨proof⟩

lemma *STAR-cos-zero* [*simp*]: (*f* cos) 0 = 1
⟨proof⟩

lemma *STAR-cos-Infinitesimal* [*simp*]:
fixes x :: 'a::{real-normed-field,banach} star
shows x ∈ *Infinitesimal* ==> (*f* cos) x ≈ 1
⟨proof⟩

lemma *STAR-tan-zero* [*simp*]: (*f* tan) 0 = 0
⟨proof⟩

lemma *STAR-tan-Infinitesimal*: x ∈ *Infinitesimal* ==> (*f* tan) x ≈ x
⟨proof⟩

lemma *STAR-sin-cos-Infinitesimal-mult*:
fixes x :: 'a::{real-normed-field,banach} star
shows x ∈ *Infinitesimal* ==> (*f* sin) x * (*f* cos) x ≈ x
⟨proof⟩

lemma *HFinite-pi*: *hypreal-of-real pi* ∈ *HFinite*
⟨*proof*⟩

lemma *lemma-split-hypreal-of-real*:
N ∈ *HNatInfinite*
==> *hypreal-of-real a* =
hypreal-of-hypnat N ∗ (*inverse*(*hypreal-of-hypnat N*) ∗ *hypreal-of-real a*)
⟨*proof*⟩

lemma *STAR-sin-Infinitesimal-divide*:
fixes *x* :: *'a*::{*real-normed-field*,*banach*} *star*
shows [|*x* ∈ *Infinitesimal*; *x* ≠ *0* |] ==> (∗*f*∗ *sin*) *x*/*x* ≈ *1*
⟨*proof*⟩

lemma *lemma-sin-pi*:
n ∈ *HNatInfinite*
==> (∗*f*∗ *sin*) (*inverse* (*hypreal-of-hypnat n*))/(*inverse* (*hypreal-of-hypnat n*)) ≈ *1*
⟨*proof*⟩

lemma *STAR-sin-inverse-HNatInfinite*:
n ∈ *HNatInfinite*
==> (∗*f*∗ *sin*) (*inverse* (*hypreal-of-hypnat n*)) ∗ *hypreal-of-hypnat n* ≈ *1*
⟨*proof*⟩

lemma *Infinitesimal-pi-divide-HNatInfinite*:
N ∈ *HNatInfinite*
==> *hypreal-of-real pi*/(*hypreal-of-hypnat N*) ∈ *Infinitesimal*
⟨*proof*⟩

lemma *pi-divide-HNatInfinite-not-zero* [*simp*]:
N ∈ *HNatInfinite* ==> *hypreal-of-real pi*/(*hypreal-of-hypnat N*) ≠ *0*
⟨*proof*⟩

lemma *STAR-sin-pi-divide-HNatInfinite-approx-pi*:
n ∈ *HNatInfinite*
==> (∗*f*∗ *sin*) (*hypreal-of-real pi*/(*hypreal-of-hypnat n*)) ∗ *hypreal-of-hypnat n*
≈ *hypreal-of-real pi*
⟨*proof*⟩

lemma *STAR-sin-pi-divide-HNatInfinite-approx-pi2*:
n ∈ *HNatInfinite*

==> *hypreal-of-hypnat n* ∗
(∗*f*∗ *sin*) (*hypreal-of-real pi*/(*hypreal-of-hypnat n*))
≈ *hypreal-of-real pi*
⟨*proof*⟩

lemma *starfunNat-pi-divide-n-Infinitesimal*:
N ∈ *HNatInfinite* ==> (∗*f*∗ (%*x*. *pi* / *real x*)) *N* ∈ *Infinitesimal*
⟨*proof*⟩

lemma *STAR-sin-pi-divide-n-approx*:
N ∈ *HNatInfinite* ==>
(∗*f*∗ *sin*) ((∗*f*∗ (%*x*. *pi* / *real x*)) *N*) ≈
hypreal-of-real pi/(*hypreal-of-hypnat N*)
⟨*proof*⟩

lemma *NSLIMSEQ-sin-pi*: (%*n*. *real n* ∗ *sin* (*pi* / *real n*)) $\longrightarrow_{NS}$ *pi*
⟨*proof*⟩

lemma *NSLIMSEQ-cos-one*: (%*n*. *cos* (*pi* / *real n*))$\longrightarrow_{NS}$ *1*
⟨*proof*⟩

lemma *NSLIMSEQ-sin-cos-pi*:
(%*n*. *real n* ∗ *sin* (*pi* / *real n*) ∗ *cos* (*pi* / *real n*)) $\longrightarrow_{NS}$ *pi*
⟨*proof*⟩

A familiar approximation to *cos x* when *x* is small

lemma *STAR-cos-Infinitesimal-approx*:
fixes *x* :: *'a*::{*real-normed-field*,*banach*} *star*
shows *x* ∈ *Infinitesimal* ==> (∗*f*∗ *cos*) *x* ≈ *1* − x^2
⟨*proof*⟩

lemma *STAR-cos-Infinitesimal-approx2*:
fixes *x* :: *hypreal* — perhaps could be generalised, like many other hypreal results

shows *x* ∈ *Infinitesimal* ==> (∗*f*∗ *cos*) *x* ≈ *1* − (x^2)/*2*
⟨*proof*⟩

end

15 Non-Standard Complex Analysis

theory *NSCA*
imports *NSComplex HTranscendental*
begin

abbreviation

SComplex :: *hcomplex set* **where**
SComplex ≡ *Standard*

definition — standard part map
stc :: *hcomplex* => *hcomplex* **where**
stc x = (*SOME r. x* ∈ *HFinite* & *r*:*SComplex* & *r* ≈ *x*)

15.1 Closure Laws for SComplex, the Standard Complex Numbers

lemma *SComplex-minus-iff* [*simp*]: (−*x* ∈ *SComplex*) = (*x* ∈ *SComplex*)
⟨*proof*⟩

lemma *SComplex-add-cancel*:
[| *x* + *y* ∈ *SComplex*; *y* ∈ *SComplex* |] ==> *x* ∈ *SComplex*
⟨*proof*⟩

lemma *SReal-hcmod-hcomplex-of-complex* [*simp*]:
hcmod (*hcomplex-of-complex r*) ∈ ℝ
⟨*proof*⟩

lemma *SReal-hcmod-numeral* [*simp*]: *hcmod* (*numeral w* ::*hcomplex*) ∈ ℝ
⟨*proof*⟩

lemma *SReal-hcmod-SComplex*: *x* ∈ *SComplex* ==> *hcmod x* ∈ ℝ
⟨*proof*⟩

lemma *SComplex-divide-numeral*:
r ∈ *SComplex* ==> *r*/(*numeral w*::*hcomplex*) ∈ *SComplex*
⟨*proof*⟩

lemma *SComplex-UNIV-complex*:
{*x*. *hcomplex-of-complex x* ∈ *SComplex*} = (*UNIV*::*complex set*)
⟨*proof*⟩

lemma *SComplex-iff*: (*x* ∈ *SComplex*) = (∃ *y*. *x* = *hcomplex-of-complex y*)
⟨*proof*⟩

lemma *hcomplex-of-complex-image*:
hcomplex-of-complex ‘(*UNIV*::*complex set*) = *SComplex*
⟨*proof*⟩

lemma *inv-hcomplex-of-complex-image*: *inv hcomplex-of-complex* ‘*SComplex* = *UNIV*
⟨*proof*⟩

lemma *SComplex-hcomplex-of-complex-image*:
[| ∃ *x*. *x*: *P*; *P* ≤ *SComplex* |] ==> ∃ *Q*. *P* = *hcomplex-of-complex* ‘ *Q*
⟨*proof*⟩

lemma *SComplex-SReal-dense*:
[| *x* ∈ *SComplex*; *y* ∈ *SComplex*; *hcmod x* < *hcmod y*

$|] \Longrightarrow \exists\, r \in \mathit{Reals}.\ \mathit{hcmod}\ x < r\ \&\ r < \mathit{hcmod}\ y$
⟨*proof*⟩

15.2 The Finite Elements form a Subring

lemma *HFinite-hcmod-hcomplex-of-complex* [*simp*]:
$\mathit{hcmod}\ (\mathit{hcomplex\text{-}of\text{-}complex}\ r) \in \mathit{HFinite}$
⟨*proof*⟩

lemma *HFinite-hcmod-iff*: $(x \in \mathit{HFinite}) = (\mathit{hcmod}\ x \in \mathit{HFinite})$
⟨*proof*⟩

lemma *HFinite-bounded-hcmod*:
$[|x \in \mathit{HFinite};\ y \le \mathit{hcmod}\ x;\ 0 \le y\ |] \Longrightarrow y\colon \mathit{HFinite}$
⟨*proof*⟩

15.3 The Complex Infinitesimals form a Subring

lemma *hcomplex-sum-of-halves*: $x/(2{::}\mathit{hcomplex}) + x/(2{::}\mathit{hcomplex}) = x$
⟨*proof*⟩

lemma *Infinitesimal-hcmod-iff*:
$(z \in \mathit{Infinitesimal}) = (\mathit{hcmod}\ z \in \mathit{Infinitesimal})$
⟨*proof*⟩

lemma *HInfinite-hcmod-iff*: $(z \in \mathit{HInfinite}) = (\mathit{hcmod}\ z \in \mathit{HInfinite})$
⟨*proof*⟩

lemma *HFinite-diff-Infinitesimal-hcmod*:
$x \in \mathit{HFinite} - \mathit{Infinitesimal} \Longrightarrow \mathit{hcmod}\ x \in \mathit{HFinite} - \mathit{Infinitesimal}$
⟨*proof*⟩

lemma *hcmod-less-Infinitesimal*:
$[|\ e \in \mathit{Infinitesimal};\ \mathit{hcmod}\ x < \mathit{hcmod}\ e\ |] \Longrightarrow x \in \mathit{Infinitesimal}$
⟨*proof*⟩

lemma *hcmod-le-Infinitesimal*:
$[|\ e \in \mathit{Infinitesimal};\ \mathit{hcmod}\ x \le \mathit{hcmod}\ e\ |] \Longrightarrow x \in \mathit{Infinitesimal}$
⟨*proof*⟩

lemma *Infinitesimal-interval-hcmod*:
$[|\ e \in \mathit{Infinitesimal};$
$e' \in \mathit{Infinitesimal};$
$\mathit{hcmod}\ e' < \mathit{hcmod}\ x\ ;\ \mathit{hcmod}\ x < \mathit{hcmod}\ e$
$|] \Longrightarrow x \in \mathit{Infinitesimal}$
⟨*proof*⟩

lemma *Infinitesimal-interval2-hcmod*:
$[|\ e \in \mathit{Infinitesimal};$
$e' \in \mathit{Infinitesimal};$

hcmod e′ ≤ hcmod x ; *hcmod x ≤ hcmod e*
|] ==> *x ∈ Infinitesimal*
⟨*proof*⟩

15.4 The "Infinitely Close" Relation

lemma *approx-SComplex-mult-cancel-zero*:
[| *a ∈ SComplex*; *a ≠ 0*; *a*∗*x ≈ 0* |] ==> *x ≈ 0*
⟨*proof*⟩

lemma *approx-mult-SComplex1*: [| *a ∈ SComplex*; *x ≈ 0* |] ==> *x*∗*a ≈ 0*
⟨*proof*⟩

lemma *approx-mult-SComplex2*: [| *a ∈ SComplex*; *x ≈ 0* |] ==> *a*∗*x ≈ 0*
⟨*proof*⟩

lemma *approx-mult-SComplex-zero-cancel-iff* [*simp*]:
[|*a ∈ SComplex*; *a ≠ 0* |] ==> (*a*∗*x ≈ 0*) = (*x ≈ 0*)
⟨*proof*⟩

lemma *approx-SComplex-mult-cancel*:
[| *a ∈ SComplex*; *a ≠ 0*; *a*∗ *w ≈ a*∗*z* |] ==> *w ≈ z*
⟨*proof*⟩

lemma *approx-SComplex-mult-cancel-iff1* [*simp*]:
[| *a ∈ SComplex*; *a ≠ 0*|] ==> (*a*∗ *w ≈ a*∗*z*) = (*w ≈ z*)
⟨*proof*⟩

lemma *approx-hcmod-approx-zero*: (*x ≈ y*) = (*hcmod* (*y − x*) *≈ 0*)
⟨*proof*⟩

lemma *approx-approx-zero-iff*: (*x ≈ 0*) = (*hcmod x ≈ 0*)
⟨*proof*⟩

lemma *approx-minus-zero-cancel-iff* [*simp*]: (*−x ≈ 0*) = (*x ≈ 0*)
⟨*proof*⟩

lemma *Infinitesimal-hcmod-add-diff*:
u ≈ 0 ==> *hcmod*(*x + u*) − *hcmod x ∈ Infinitesimal*
⟨*proof*⟩

lemma *approx-hcmod-add-hcmod*: *u ≈ 0* ==> *hcmod*(*x + u*) *≈ hcmod x*
⟨*proof*⟩

15.5 Zero is the Only Infinitesimal Complex Number

lemma *Infinitesimal-less-SComplex*:
[| *x ∈ SComplex*; *y ∈ Infinitesimal*; *0 < hcmod x* |] ==> *hcmod y < hcmod x*

⟨*proof*⟩

lemma *SComplex-Int-Infinitesimal-zero*: *SComplex Int Infinitesimal* = {*0*}
⟨*proof*⟩

lemma *SComplex-Infinitesimal-zero*:
[| $x \in$ *SComplex*; $x \in$ *Infinitesimal*|] ==> $x = 0$
⟨*proof*⟩

lemma *SComplex-HFinite-diff-Infinitesimal*:
[| $x \in$ *SComplex*; $x \neq 0$ |] ==> $x \in$ *HFinite* − *Infinitesimal*
⟨*proof*⟩

lemma *hcomplex-of-complex-HFinite-diff-Infinitesimal*:
hcomplex-of-complex $x \neq 0$
==> *hcomplex-of-complex* $x \in$ *HFinite* − *Infinitesimal*
⟨*proof*⟩

lemma *numeral-not-Infinitesimal* [*simp*]:
numeral $w \neq$ (*0*::*hcomplex*) ==> (*numeral* w::*hcomplex*) $\notin$ *Infinitesimal*
⟨*proof*⟩

lemma *approx-SComplex-not-zero*:
[| $y \in$ *SComplex*; $x \approx y$; $y \neq 0$ |] ==> $x \neq 0$
⟨*proof*⟩

lemma *SComplex-approx-iff*:
[|$x \in$ *SComplex*; $y \in$ *SComplex*|] ==> $(x \approx y) = (x = y)$
⟨*proof*⟩

lemma *numeral-Infinitesimal-iff* [*simp*]:
((*numeral* w :: *hcomplex*) $\in$ *Infinitesimal*) =
(*numeral* w = (*0*::*hcomplex*))
⟨*proof*⟩

lemma *approx-unique-complex*:
[| $r \in$ *SComplex*; $s \in$ *SComplex*; $r \approx x$; $s \approx x$|] ==> $r = s$
⟨*proof*⟩

15.6 Properties of *hRe*, *hIm* and *HComplex*

lemma *abs-hRe-le-hcmod*: $\bigwedge x.$ |*hRe* x| $\leq$ *hcmod* x
⟨*proof*⟩

lemma *abs-hIm-le-hcmod*: $\bigwedge x.$ |*hIm* x| $\leq$ *hcmod* x
⟨*proof*⟩

lemma *Infinitesimal-hRe*: $x \in$ *Infinitesimal* $\Longrightarrow$ *hRe* $x \in$ *Infinitesimal*
⟨*proof*⟩

lemma *Infinitesimal-hIm*: $x \in \mathit{Infinitesimal} \Longrightarrow \mathit{hIm}\ x \in \mathit{Infinitesimal}$
⟨*proof*⟩

lemma *real-sqrt-lessI*: $[\![0 < u;\ x < u^2]\!] \Longrightarrow \mathit{sqrt}\ x < u$

⟨*proof*⟩

lemma *hypreal-sqrt-lessI*:
$\bigwedge x\ u.\ [\![0 < u;\ x < u^2]\!] \Longrightarrow (\ {*}f{*}\ \mathit{sqrt})\ x < u$
⟨*proof*⟩

lemma *hypreal-sqrt-ge-zero*: $\bigwedge x.\ 0 \le x \Longrightarrow 0 \le (\ {*}f{*}\ \mathit{sqrt})\ x$
⟨*proof*⟩

lemma *Infinitesimal-sqrt*:
$[\![x \in \mathit{Infinitesimal};\ 0 \le x]\!] \Longrightarrow (\ {*}f{*}\ \mathit{sqrt})\ x \in \mathit{Infinitesimal}$
⟨*proof*⟩

lemma *Infinitesimal-HComplex*:
$[\![x \in \mathit{Infinitesimal};\ y \in \mathit{Infinitesimal}]\!] \Longrightarrow \mathit{HComplex}\ x\ y \in \mathit{Infinitesimal}$
⟨*proof*⟩

lemma *hcomplex-Infinitesimal-iff*:
$(x \in \mathit{Infinitesimal}) = (\mathit{hRe}\ x \in \mathit{Infinitesimal} \wedge \mathit{hIm}\ x \in \mathit{Infinitesimal})$
⟨*proof*⟩

lemma *hRe-diff* [*simp*]: $\bigwedge x\ y.\ \mathit{hRe}\ (x - y) = \mathit{hRe}\ x - \mathit{hRe}\ y$
⟨*proof*⟩

lemma *hIm-diff* [*simp*]: $\bigwedge x\ y.\ \mathit{hIm}\ (x - y) = \mathit{hIm}\ x - \mathit{hIm}\ y$
⟨*proof*⟩

lemma *approx-hRe*: $x \approx y \Longrightarrow \mathit{hRe}\ x \approx \mathit{hRe}\ y$
⟨*proof*⟩

lemma *approx-hIm*: $x \approx y \Longrightarrow \mathit{hIm}\ x \approx \mathit{hIm}\ y$
⟨*proof*⟩

lemma *approx-HComplex*:
$[\![a \approx b;\ c \approx d]\!] \Longrightarrow \mathit{HComplex}\ a\ c \approx \mathit{HComplex}\ b\ d$
⟨*proof*⟩

lemma *hcomplex-approx-iff*:
$(x \approx y) = (\mathit{hRe}\ x \approx \mathit{hRe}\ y \wedge \mathit{hIm}\ x \approx \mathit{hIm}\ y)$
⟨*proof*⟩

lemma *HFinite-hRe*: $x \in \mathit{HFinite} \Longrightarrow \mathit{hRe}\ x \in \mathit{HFinite}$
⟨*proof*⟩

lemma *HFinite-hIm*: $x \in HFinite \Longrightarrow hIm\ x \in HFinite$
⟨*proof*⟩

lemma *HFinite-HComplex*:
$[\![x \in HFinite; y \in HFinite]\!] \Longrightarrow HComplex\ x\ y \in HFinite$
⟨*proof*⟩

lemma *hcomplex-HFinite-iff*:
$(x \in HFinite) = (hRe\ x \in HFinite \wedge hIm\ x \in HFinite)$
⟨*proof*⟩

lemma *hcomplex-HInfinite-iff*:
$(x \in HInfinite) = (hRe\ x \in HInfinite \vee hIm\ x \in HInfinite)$
⟨*proof*⟩

lemma *hcomplex-of-hypreal-approx-iff* [*simp*]:
$(hcomplex\text{-}of\text{-}hypreal\ x \approx hcomplex\text{-}of\text{-}hypreal\ z) = (x \approx z)$
⟨*proof*⟩

lemma *Standard-HComplex*:
$[\![x \in Standard; y \in Standard]\!] \Longrightarrow HComplex\ x\ y \in Standard$
⟨*proof*⟩

lemma *stc-part-Ex*: $x{:}HFinite$ ==> $\exists\, t \in SComplex.\ x \approx t$
⟨*proof*⟩

lemma *stc-part-Ex1*: $x{:}HFinite$ ==> $\exists !t.\ t \in SComplex$ & $x \approx t$
⟨*proof*⟩

lemmas *hcomplex-of-complex-approx-inverse* =
hcomplex-of-complex-HFinite-diff-Infinitesimal [*THEN* [*2*] *approx-inverse*]

15.7 Theorems About Monads

lemma *monad-zero-hcmod-iff*: $(x \in monad\ 0) = (hcmod\ x{:}monad\ 0)$
⟨*proof*⟩

15.8 Theorems About Standard Part

lemma *stc-approx-self*: $x \in HFinite$ ==> $stc\ x \approx x$
⟨*proof*⟩

lemma *stc-SComplex*: $x \in HFinite$ ==> $stc\ x \in SComplex$
⟨*proof*⟩

lemma *stc-HFinite*: $x \in HFinite$ ==> $stc\ x \in HFinite$
⟨*proof*⟩

lemma *stc-unique*: $[\![y \in SComplex;\ y \approx x]\!] \Longrightarrow stc\ x = y$
⟨*proof*⟩

lemma *stc-SComplex-eq* [*simp*]: $x \in SComplex$ ==> $stc\ x = x$
⟨*proof*⟩

lemma *stc-hcomplex-of-complex*:
stc (*hcomplex-of-complex* x) = *hcomplex-of-complex* x
⟨*proof*⟩

lemma *stc-eq-approx*:
[| $x \in HFinite$; $y \in HFinite$; $stc\ x = stc\ y$ |] ==> $x \approx y$
⟨*proof*⟩

lemma *approx-stc-eq*:
[| $x \in HFinite$; $y \in HFinite$; $x \approx y$ |] ==> $stc\ x = stc\ y$
⟨*proof*⟩

lemma *stc-eq-approx-iff*:
[| $x \in HFinite$; $y \in HFinite$|] ==> $(x \approx y) = (stc\ x = stc\ y)$
⟨*proof*⟩

lemma *stc-Infinitesimal-add-SComplex*:
[| $x \in SComplex$; $e \in Infinitesimal$ |] ==> $stc(x + e) = x$
⟨*proof*⟩

lemma *stc-Infinitesimal-add-SComplex2*:
[| $x \in SComplex$; $e \in Infinitesimal$ |] ==> $stc(e + x) = x$
⟨*proof*⟩

lemma *HFinite-stc-Infinitesimal-add*:
$x \in HFinite$ ==> $\exists\, e \in Infinitesimal.\ x = stc(x) + e$
⟨*proof*⟩

lemma *stc-add*:
[| $x \in HFinite$; $y \in HFinite$ |] ==> $stc\ (x + y) = stc(x) + stc(y)$
⟨*proof*⟩

lemma *stc-numeral* [*simp*]: *stc* (*numeral* w) = *numeral* w
⟨*proof*⟩

lemma *stc-zero* [*simp*]: $stc\ 0 = 0$
⟨*proof*⟩

lemma *stc-one* [*simp*]: $stc\ 1 = 1$
⟨*proof*⟩

lemma *stc-minus*: $y \in HFinite$ ==> $stc(-y) = -stc(y)$
⟨*proof*⟩

lemma *stc-diff*:
$[| \, x \in HFinite; \; y \in HFinite \, |] \Longrightarrow stc \; (x - y) = stc(x) - stc(y)$
⟨*proof*⟩

lemma *stc-mult*:
$[| \, x \in HFinite; \; y \in HFinite \, |]$
$\Longrightarrow stc \; (x * y) = stc(x) * stc(y)$
⟨*proof*⟩

lemma *stc-Infinitesimal*: $x \in Infinitesimal \Longrightarrow stc \; x = 0$
⟨*proof*⟩

lemma *stc-not-Infinitesimal*: $stc(x) \neq 0 \Longrightarrow x \notin Infinitesimal$
⟨*proof*⟩

lemma *stc-inverse*:
$[| \, x \in HFinite; \; stc \; x \neq 0 \, |]$
$\Longrightarrow stc(inverse \; x) = inverse \; (stc \; x)$
⟨*proof*⟩

lemma *stc-divide* [*simp*]:
$[| \, x \in HFinite; \; y \in HFinite; \; stc \; y \neq 0 \, |]$
$\Longrightarrow stc(x/y) = (stc \; x) \; / \; (stc \; y)$
⟨*proof*⟩

lemma *stc-idempotent* [*simp*]: $x \in HFinite \Longrightarrow stc(stc(x)) = stc(x)$
⟨*proof*⟩

lemma *HFinite-HFinite-hcomplex-of-hypreal*:
$z \in HFinite \Longrightarrow hcomplex\text{-}of\text{-}hypreal \; z \in HFinite$
⟨*proof*⟩

lemma *SComplex-SReal-hcomplex-of-hypreal*:
$x \in \mathbb{R} \Longrightarrow hcomplex\text{-}of\text{-}hypreal \; x \in SComplex$
⟨*proof*⟩

lemma *stc-hcomplex-of-hypreal*:
$z \in HFinite \Longrightarrow stc(hcomplex\text{-}of\text{-}hypreal \; z) = hcomplex\text{-}of\text{-}hypreal \; (st \; z)$
⟨*proof*⟩

lemma *Infinitesimal-hcnj-iff* [*simp*]:
$(hcnj \; z \in Infinitesimal) = (z \in Infinitesimal)$
⟨*proof*⟩

lemma *Infinitesimal-hcomplex-of-hypreal-epsilon* [*simp*]:
$hcomplex\text{-}of\text{-}hypreal \; \varepsilon \in Infinitesimal$

⟨*proof*⟩

end

16 Star-transforms in NSA, Extending Sets of Complex Numbers and Complex Functions

theory *CStar*
imports *NSCA*
begin

16.1 Properties of the ∗-Transform Applied to Sets of Reals

lemma *STARC-hcomplex-of-complex-Int*: $*s*\ X \cap SComplex = hcomplex\text{-}of\text{-}complex$ ‘ X
⟨*proof*⟩

lemma *lemma-not-hcomplexA*: $x \notin hcomplex\text{-}of\text{-}complex$ ‘ $A \Longrightarrow \forall\, y \in A.\ x \neq hcomplex\text{-}of\text{-}complex\ y$
⟨*proof*⟩

16.2 Theorems about Nonstandard Extensions of Functions

lemma *starfunC-hcpow*: $\bigwedge Z.\ (\ *f*\ (\lambda z.\ z\ \hat{}\ n))\ Z = Z\ pow\ hypnat\text{-}of\text{-}nat\ n$
⟨*proof*⟩

lemma *starfunCR-cmod*: $*f*\ cmod = hcmod$
⟨*proof*⟩

16.3 Internal Functions - Some Redundancy With ∗f∗ Now

lemma *starfun-Re*: $(\ *f*\ (\lambda x.\ Re\ (f\,x))) = (\lambda x.\ hRe\ ((\ *f*\ f)\ x))$
⟨*proof*⟩

lemma *starfun-Im*: $(\ *f*\ (\lambda x.\ Im\ (f\,x))) = (\lambda x.\ hIm\ ((\ *f*\ f)\ x))$
⟨*proof*⟩

lemma *starfunC-eq-Re-Im-iff*:
$(\ *f*\ f)\ x = z \longleftrightarrow (\ *f*\ (\lambda x.\ Re\ (f\,x)))\ x = hRe\ z \wedge (\ *f*\ (\lambda x.\ Im\ (f\,x)))\ x = hIm\ z$
⟨*proof*⟩

lemma *starfunC-approx-Re-Im-iff*:
$(\ *f*\ f)\ x \approx z \longleftrightarrow (\ *f*\ (\lambda x.\ Re\ (f\,x)))\ x \approx hRe\ z \wedge (\ *f*\ (\lambda x.\ Im\ (f\,x)))\ x \approx hIm\ z$
⟨*proof*⟩

end

17 Limits, Continuity and Differentiation for Complex Functions

theory *CLim*
 imports *CStar*
begin

declare *hypreal-epsilon-not-zero* [*simp*]

lemma *lemma-complex-mult-inverse-squared* [*simp*]: $x \neq 0 \Longrightarrow x * (inverse\ x)^2 = inverse\ x$
 for x :: *complex*
 ⟨*proof*⟩

Changing the quantified variable. Install earlier?

lemma *all-shift*: $(\forall x::'a::comm\text{-}ring\text{-}1.\ P\ x) \longleftrightarrow (\forall x.\ P\ (x - a))$
 ⟨*proof*⟩

lemma *complex-add-minus-iff* [*simp*]: $x + - a = 0 \longleftrightarrow x = a$
 for $x\ a$ :: *complex*
 ⟨*proof*⟩

lemma *complex-add-eq-0-iff* [*iff*]: $x + y = 0 \longleftrightarrow y = - x$
 for $x\ y$ :: *complex*
 ⟨*proof*⟩

17.1 Limit of Complex to Complex Function

lemma *NSLIM-Re*: $f\ -a\rightarrow_{NS}\ L \Longrightarrow (\lambda x.\ Re\ (f\ x))\ -a\rightarrow_{NS}\ Re\ L$
 ⟨*proof*⟩

lemma *NSLIM-Im*: $f\ -a\rightarrow_{NS}\ L \Longrightarrow (\lambda x.\ Im\ (f\ x))\ -a\rightarrow_{NS}\ Im\ L$
 ⟨*proof*⟩

lemma *LIM-Re*: $f\ -a\rightarrow\ L \Longrightarrow (\lambda x.\ Re\ (f\ x))\ -a\rightarrow\ Re\ L$
 for f :: *'a::real-normed-vector* ⇒ *complex*
 ⟨*proof*⟩

lemma *LIM-Im*: $f\ -a\rightarrow\ L \Longrightarrow (\lambda x.\ Im\ (f\ x))\ -a\rightarrow\ Im\ L$
 for f :: *'a::real-normed-vector* ⇒ *complex*
 ⟨*proof*⟩

lemma *LIM-cnj*: $f\ -a\rightarrow\ L \Longrightarrow (\lambda x.\ cnj\ (f\ x))\ -a\rightarrow\ cnj\ L$
 for f :: *'a::real-normed-vector* ⇒ *complex*
 ⟨*proof*⟩

lemma *LIM-cnj-iff*: $((\lambda x.\ cnj\ (f\ x))\ -a\rightarrow\ cnj\ L) \longleftrightarrow f\ -a\rightarrow\ L$

for *f* :: *'a::real-normed-vector* ⇒ *complex*
⟨*proof*⟩

lemma *starfun-norm*: $(*f* (\lambda x.\ norm\ (f\ x))) = (\lambda x.\ hnorm\ ((*f* f)\ x))$
⟨*proof*⟩

lemma *star-of-Re* [*simp*]: $star\text{-}of\ (Re\ x) = hRe\ (star\text{-}of\ x)$
⟨*proof*⟩

lemma *star-of-Im* [*simp*]: $star\text{-}of\ (Im\ x) = hIm\ (star\text{-}of\ x)$
⟨*proof*⟩

Another equivalence result.

lemma *NSCLIM-NSCRLIM-iff*: $f\ -x\rightarrow_{NS}\ L \longleftrightarrow (\lambda y.\ cmod\ (f\ y - L))\ -x\rightarrow_{NS}$ 0
⟨*proof*⟩

Much, much easier standard proof.

lemma *CLIM-CRLIM-iff*: $f\ -x\rightarrow\ L \longleftrightarrow (\lambda y.\ cmod\ (f\ y - L))\ -x\rightarrow\ 0$
for *f* :: *'a::real-normed-vector* ⇒ *complex*
⟨*proof*⟩

So this is nicer nonstandard proof.

lemma *NSCLIM-NSCRLIM-iff2*: $f\ -x\rightarrow_{NS}\ L \longleftrightarrow (\lambda y.\ cmod\ (f\ y - L))\ -x\rightarrow_{NS}$ 0
⟨*proof*⟩

lemma *NSLIM-NSCRLIM-Re-Im-iff*:
$f\ -a\rightarrow_{NS}\ L \longleftrightarrow (\lambda x.\ Re\ (f\ x))\ -a\rightarrow_{NS}\ Re\ L \land (\lambda x.\ Im\ (f\ x))\ -a\rightarrow_{NS}\ Im\ L$
⟨*proof*⟩

lemma *LIM-CRLIM-Re-Im-iff*: $f\ -a\rightarrow\ L \longleftrightarrow (\lambda x.\ Re\ (f\ x))\ -a\rightarrow\ Re\ L \land (\lambda x.\ Im\ (f\ x))\ -a\rightarrow\ Im\ L$
for *f* :: *'a::real-normed-vector* ⇒ *complex*
⟨*proof*⟩

17.2 Continuity

lemma *NSLIM-isContc-iff*: $f\ -a\rightarrow_{NS}\ f\ a \longleftrightarrow (\lambda h.\ f\ (a + h))\ -0\rightarrow_{NS}\ f\ a$
⟨*proof*⟩

17.3 Functions from Complex to Reals

lemma *isNSContCR-cmod* [*simp*]: *isNSCont cmod a*
⟨*proof*⟩

lemma *isContCR-cmod* [*simp*]: *isCont cmod a*
⟨*proof*⟩

lemma *isCont-Re*: *isCont f a* $\Longrightarrow$ *isCont* (λx. *Re* (*f x*)) *a*
for *f* :: *'a::real-normed-vector* $\Rightarrow$ *complex*
⟨*proof*⟩

lemma *isCont-Im*: *isCont f a* $\Longrightarrow$ *isCont* (λx. *Im* (*f x*)) *a*
for *f* :: *'a::real-normed-vector* $\Rightarrow$ *complex*
⟨*proof*⟩

17.4 Differentiation of Natural Number Powers

lemma *CDERIV-pow* [*simp*]: *DERIV* (λx. *x* ^ *n*) *x* :> *complex-of-real* (*real n*) $*$ (*x* ^ (*n* $-$ *Suc 0*))
⟨*proof*⟩

Nonstandard version.

lemma *NSCDERIV-pow*: *NSDERIV* (λx. *x* ^ *n*) *x* :> *complex-of-real* (*real n*) $*$ (*x* ^ (*n* $-$ *1*))
⟨*proof*⟩

Can't relax the premise $x \neq (0::'a)$: it isn't continuous at zero.

lemma *NSCDERIV-inverse*: $x \neq 0 \Longrightarrow$ *NSDERIV* (λx. *inverse x*) *x* :> $-$ (*inverse x*)^2
for *x* :: *complex*
⟨*proof*⟩

lemma *CDERIV-inverse*: $x \neq 0 \Longrightarrow$ *DERIV* (λx. *inverse x*) *x* :> $-$ (*inverse x*)^2
for *x* :: *complex*
⟨*proof*⟩

17.5 Derivative of Reciprocals (Function *inverse*)

lemma *CDERIV-inverse-fun*:
DERIV f x :> *d* $\Longrightarrow$ *f x* $\neq 0 \Longrightarrow$ *DERIV* (λx. *inverse* (*f x*)) *x* :> $-$ (*d* $*$ *inverse* ((*f x*)^2))
for *x* :: *complex*
⟨*proof*⟩

lemma *NSCDERIV-inverse-fun*:
NSDERIV f x :> *d* $\Longrightarrow$ *f x* $\neq 0 \Longrightarrow$ *NSDERIV* (λx. *inverse* (*f x*)) *x* :> $-$ (*d* $*$ *inverse* ((*f x*)^2))
for *x* :: *complex*
⟨*proof*⟩

17.6 Derivative of Quotient

lemma *CDERIV-quotient*:
DERIV f x :> *d* $\Longrightarrow$ *DERIV g x* :> *e* $\Longrightarrow$ $g(x) \neq 0 \Longrightarrow$
DERIV (λy. *f y* / *g y*) *x* :> (*d* $*$ *g x* $-$ (*e* $*$ *f x*)) / (*g x*)^2
for *x* :: *complex*

⟨*proof*⟩

lemma *NSCDERIV-quotient*:
NSDERIV f x :> d $\Longrightarrow$ *NSDERIV g x :> e* $\Longrightarrow$ *g x* $\neq$ *(0::complex)* $\Longrightarrow$
NSDERIV $(\lambda y.\ f\ y\ /\ g\ y)\ x :> (d * g\ x - (e * f\ x))\ /\ (g\ x)^2$
⟨*proof*⟩

17.7 Caratheodory Formulation of Derivative at a Point: Standard Proof

lemma *CARAT-CDERIVD*:
$(\forall z.\ f\ z - f\ x = g\ z * (z - x)) \wedge$ *isNSCont g x* $\wedge\ g\ x = l \Longrightarrow$ *NSDERIV f x :>*
l
⟨*proof*⟩

end

18 Logarithms: Non-Standard Version

theory *HLog*
imports *HTranscendental*
begin

lemma *epsilon-ge-zero* [*simp*]: $0 \leq \varepsilon$
⟨*proof*⟩

lemma *hpfinite-witness*: $\varepsilon \in \{x.\ 0 \leq x \wedge x \in$ *HFinite*$\}$
⟨*proof*⟩

definition *powhr* :: *hypreal* $\Rightarrow$ *hypreal* $\Rightarrow$ *hypreal* (**infixr** *powhr 80*)
where [*transfer-unfold*]: *x powhr a = starfun2 (op powr) x a*

definition *hlog* :: *hypreal* $\Rightarrow$ *hypreal* $\Rightarrow$ *hypreal*
where [*transfer-unfold*]: *hlog a x = starfun2 log a x*

lemma *powhr*: (*star-n X*) *powhr* (*star-n Y*) = *star-n* (λn. (*X n*) *powr* (*Y n*))
⟨*proof*⟩

lemma *powhr-one-eq-one* [*simp*]: $\bigwedge a$. *1 powhr a = 1*
⟨*proof*⟩

lemma *powhr-mult*: $\bigwedge a\ x\ y.\ 0 < x \Longrightarrow 0 < y \Longrightarrow (x * y)$ *powhr a* = (*x powhr*
a) $*$ (*y powhr a*)
⟨*proof*⟩

lemma *powhr-gt-zero* [*simp*]: $\bigwedge a\ x.\ 0 < x$ *powhr a* $\longleftrightarrow x \neq 0$

⟨*proof*⟩

lemma *powhr-not-zero* [*simp*]: $\bigwedge a\ x.\ x\ powhr\ a \neq 0 \longleftrightarrow x \neq 0$
⟨*proof*⟩

lemma *powhr-divide*: $\bigwedge a\ x\ y.\ 0 < x \Longrightarrow 0 < y \Longrightarrow (x\ /\ y)\ powhr\ a = (x\ powhr\ a)\ /\ (y\ powhr\ a)$
⟨*proof*⟩

lemma *powhr-add*: $\bigwedge a\ b\ x.\ x\ powhr\ (a + b) = (x\ powhr\ a) * (x\ powhr\ b)$
⟨*proof*⟩

lemma *powhr-powhr*: $\bigwedge a\ b\ x.\ (x\ powhr\ a)\ powhr\ b = x\ powhr\ (a * b)$
⟨*proof*⟩

lemma *powhr-powhr-swap*: $\bigwedge a\ b\ x.\ (x\ powhr\ a)\ powhr\ b = (x\ powhr\ b)\ powhr\ a$
⟨*proof*⟩

lemma *powhr-minus*: $\bigwedge a\ x.\ x\ powhr\ (-\ a) = inverse\ (x\ powhr\ a)$
⟨*proof*⟩

lemma *powhr-minus-divide*: $x\ powhr\ (-\ a) = 1\ /\ (x\ powhr\ a)$
⟨*proof*⟩

lemma *powhr-less-mono*: $\bigwedge a\ b\ x.\ a < b \Longrightarrow 1 < x \Longrightarrow x\ powhr\ a < x\ powhr\ b$
⟨*proof*⟩

lemma *powhr-less-cancel*: $\bigwedge a\ b\ x.\ x\ powhr\ a < x\ powhr\ b \Longrightarrow 1 < x \Longrightarrow a < b$
⟨*proof*⟩

lemma *powhr-less-cancel-iff* [*simp*]: $1 < x \Longrightarrow x\ powhr\ a < x\ powhr\ b \longleftrightarrow a < b$
⟨*proof*⟩

lemma *powhr-le-cancel-iff* [*simp*]: $1 < x \Longrightarrow x\ powhr\ a \leq x\ powhr\ b \longleftrightarrow a \leq b$
⟨*proof*⟩

lemma *hlog*: $hlog\ (star\text{-}n\ X)\ (star\text{-}n\ Y) = star\text{-}n\ (\lambda n.\ log\ (X\ n)\ (Y\ n))$
⟨*proof*⟩

lemma *hlog-starfun-ln*: $\bigwedge x.\ (\ {*}f{*}\ ln)\ x = hlog\ ((\ {*}f{*}\ exp)\ 1)\ x$
⟨*proof*⟩

lemma *powhr-hlog-cancel* [*simp*]: $\bigwedge a\ x.\ 0 < a \Longrightarrow a \neq 1 \Longrightarrow 0 < x \Longrightarrow a\ powhr\ (hlog\ a\ x) = x$
⟨*proof*⟩

lemma *hlog-powhr-cancel* [*simp*]: $\bigwedge a\ y.\ 0 < a \Longrightarrow a \neq 1 \Longrightarrow hlog\ a\ (a\ powhr\ y) = y$
⟨*proof*⟩

lemma *hlog-mult*:
$\bigwedge a\ x\ y.\ 0 < a \Longrightarrow a \neq 1 \Longrightarrow 0 < x \Longrightarrow 0 < y \Longrightarrow hlog\ a\ (x * y) = hlog\ a\ x + hlog\ a\ y$
⟨*proof*⟩

lemma *hlog-as-starfun*: $\bigwedge a\ x.\ 0 < a \Longrightarrow a \neq 1 \Longrightarrow hlog\ a\ x = (\ {*f*}\ ln)\ x\ /\ (\ {*f*}\ ln)\ a$
⟨*proof*⟩

lemma *hlog-eq-div-starfun-ln-mult-hlog*:
$\bigwedge a\ b\ x.\ 0 < a \Longrightarrow a \neq 1 \Longrightarrow 0 < b \Longrightarrow b \neq 1 \Longrightarrow 0 < x \Longrightarrow$
$hlog\ a\ x = ((\ {*f*}\ ln)\ b\ /\ (\ {*f*}\ ln)\ a) * hlog\ b\ x$
⟨*proof*⟩

lemma *powhr-as-starfun*: $\bigwedge a\ x.\ x\ powhr\ a = (if\ x = 0\ then\ 0\ else\ (\ {*f*}\ exp)\ (a * (\ {*f*}\ real\text{-}ln)\ x))$
⟨*proof*⟩

lemma *HInfinite-powhr*:
$x \in HInfinite \Longrightarrow 0 < x \Longrightarrow a \in HFinite - Infinitesimal \Longrightarrow 0 < a \Longrightarrow x\ powhr\ a \in HInfinite$
⟨*proof*⟩

lemma *hlog-hrabs-HInfinite-Infinitesimal*:
$x \in HFinite - Infinitesimal \Longrightarrow a \in HInfinite \Longrightarrow 0 < a \Longrightarrow hlog\ a\ |x| \in Infinitesimal$
⟨*proof*⟩

lemma *hlog-HInfinite-as-starfun*: $a \in HInfinite \Longrightarrow 0 < a \Longrightarrow hlog\ a\ x = (\ {*f*}\ ln)\ x\ /\ (\ {*f*}\ ln)\ a$
⟨*proof*⟩

lemma *hlog-one* [*simp*]: $\bigwedge a.\ hlog\ a\ 1 = 0$
⟨*proof*⟩

lemma *hlog-eq-one* [*simp*]: $\bigwedge a.\ 0 < a \Longrightarrow a \neq 1 \Longrightarrow hlog\ a\ a = 1$
⟨*proof*⟩

lemma *hlog-inverse*: $0 < a \Longrightarrow a \neq 1 \Longrightarrow 0 < x \Longrightarrow hlog\ a\ (inverse\ x) = -\ hlog\ a\ x$
⟨*proof*⟩

lemma *hlog-divide*: $0 < a \Longrightarrow a \neq 1 \Longrightarrow 0 < x \Longrightarrow 0 < y \Longrightarrow hlog\ a\ (x\ /\ y) = hlog\ a\ x - hlog\ a\ y$
⟨*proof*⟩

lemma *hlog-less-cancel-iff* [*simp*]:
$\bigwedge a\ x\ y.\ 1 < a \Longrightarrow 0 < x \Longrightarrow 0 < y \Longrightarrow hlog\ a\ x < hlog\ a\ y \longleftrightarrow x < y$

⟨*proof*⟩

lemma *hlog-le-cancel-iff* [*simp*]: $1 < a \Longrightarrow 0 < x \Longrightarrow 0 < y \Longrightarrow hlog\ a\ x \leq hlog\ a\ y \longleftrightarrow x \leq y$
⟨*proof*⟩

end

theory *Hyperreal*
imports *HLog*
begin

end
theory *Hypercomplex*
imports *CLim Hyperreal*
begin

end

theory *Nonstandard-Analysis*
imports *Hypercomplex*
begin

end